

FflasFpack

Generated by Doxygen 1.9.1



<b>1 FFLAS-FFPACK Documentation.</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Goals . . . . .	1
1.3 Design . . . . .	1
1.4 Using FFLAS-FFPACK. . . . .	1
1.5 Contributing to fflas-ffpack, getting assistance. . . . .	1
<b>2 Configuring and Installing FFLAS-FFPACK</b>	<b>3</b>
<b>3 Copying and Licence</b>	<b>5</b>
<b>4 Tutorial</b>	<b>7</b>
<b>5 Architecture of the library.</b>	<b>9</b>
<b>6 Bug List</b>	<b>11</b>
<b>7 Bibliography</b>	<b>13</b>
<b>8 Todo List</b>	<b>15</b>
<b>9 Module Index</b>	<b>17</b>
9.1 Modules . . . . .	17
<b>10 Namespace Index</b>	<b>19</b>
10.1 Namespace List . . . . .	19
<b>11 Hierarchical Index</b>	<b>21</b>
11.1 Class Hierarchy . . . . .	21
<b>12 Data Structure Index</b>	<b>29</b>
12.1 Data Structures . . . . .	29
<b>13 File Index</b>	<b>37</b>
13.1 File List . . . . .	37
<b>14 Module Documentation</b>	<b>45</b>
14.1 CHECKER . . . . .	45
14.2 FFLAS . . . . .	45
14.3 Matrix Multiplication Algorithms . . . . .	45
14.3.1 Detailed Description . . . . .	45
14.4 SIMD wrapper . . . . .	46
14.5 FFLAS-FFPACK . . . . .	46
14.5.1 Detailed Description . . . . .	46
14.6 FFPACK . . . . .	46
14.7 FFLAS-FFPACK fields . . . . .	46
14.7.1 Detailed Description . . . . .	47

14.8 RNS	47
14.9 Interfaces	47
<b>15 Namespace Documentation</b>	<b>49</b>
15.1 FFLAS Namespace Reference	49
15.1.1 Typedef Documentation	76
15.1.1.1 Checker_fgemm	76
15.1.1.2 Checker_ftrsm	76
15.1.1.3 ForceCheck_fgemm	76
15.1.1.4 ForceCheck_ftrsm	76
15.1.1.5 ZOSparseMatrix	76
15.1.1.6 NotZOSparseMatrix	76
15.1.1.7 SimdSparseMatrix	77
15.1.1.8 NoSimdSparseMatrix	77
15.1.1.9 MKLSparseMatrixFormat	77
15.1.1.10 NotMKLSparseMatrixFormat	77
15.1.1.11 has_plus	77
15.1.1.12 has_minus	77
15.1.1.13 has_equal	77
15.1.1.14 has_plus_eq	78
15.1.1.15 has_minus_eq	78
15.1.1.16 has_mul	78
15.1.1.17 has_mul_eq	78
15.1.1.18 Timer	78
15.1.1.19 BaseTimer	78
15.1.1.20 UserTimer	78
15.1.1.21 SysTimer	79
15.1.2 Enumeration Type Documentation	79
15.1.2.1 FFLAS_ORDER	79
15.1.2.2 FFLAS_TRANSPOSE	80
15.1.2.3 FFLAS_UPLO	80
15.1.2.4 FFLAS_DIAG	80
15.1.2.5 FFLAS_SIDE	81
15.1.2.6 FFLAS_BASE	81
15.1.2.7 number_kind	81
15.1.2.8 SparseMatrix_t	81
15.1.2.9 FFLAS_FORMAT	82
15.1.3 Function Documentation	82
15.1.3.1 InfNorm()	82
15.1.3.2 min3()	83
15.1.3.3 max3()	83
15.1.3.4 min4()	83

15.1.3.5 max4()	83
15.1.3.6 fadd() [1/8]	83
15.1.3.7 faddin() [1/5]	84
15.1.3.8 fsub() [1/4]	84
15.1.3.9 fsubin() [1/3]	84
15.1.3.10 fadd() [2/8]	84
15.1.3.11 pfadd()	85
15.1.3.12 pfsub()	85
15.1.3.13 pfaddin()	85
15.1.3.14 pfsubin()	85
15.1.3.15 fadd() [3/8]	86
15.1.3.16 fsub() [2/4]	86
15.1.3.17 faddin() [2/5]	87
15.1.3.18 faddin() [3/5]	87
15.1.3.19 fsubin() [2/3]	87
15.1.3.20 fadd() [4/8]	88
15.1.3.21 fassign() [1/10]	88
15.1.3.22 fassign() [2/10]	89
15.1.3.23 fassign() [3/10]	89
15.1.3.24 fassign() [4/10]	89
15.1.3.25 fassign() [5/10]	90
15.1.3.26 fassign() [6/10]	90
15.1.3.27 fassign() [7/10]	90
15.1.3.28 fassign() [8/10]	90
15.1.3.29 faxpy() [1/6]	91
15.1.3.30 faxpy() [2/6]	91
15.1.3.31 faxpy() [3/6]	91
15.1.3.32 faxpy() [4/6]	92
15.1.3.33 fdot() [1/11]	92
15.1.3.34 fdot() [2/11]	92
15.1.3.35 fdot() [3/11]	93
15.1.3.36 fdot() [4/11]	93
15.1.3.37 fdot() [5/11]	93
15.1.3.38 fdot() [6/11]	93
15.1.3.39 fdot() [7/11]	94
15.1.3.40 fdot() [8/11]	94
15.1.3.41 fgemm() [1/23]	94
15.1.3.42 fgemm() [2/23]	95
15.1.3.43 fgemm() [3/23]	95
15.1.3.44 fgemm() [4/23]	96
15.1.3.45 fgemm() [5/23]	96
15.1.3.46 fgemm() [6/23]	96

15.1.3.47 <code>fgemm()</code> [7/23]	97
15.1.3.48 <code>fgemm()</code> [8/23]	97
15.1.3.49 <code>fgemm()</code> [9/23]	98
15.1.3.50 <code>fgemm()</code> [10/23]	98
15.1.3.51 <code>fgemm()</code> [11/23]	98
15.1.3.52 <code>fgemm()</code> [12/23]	99
15.1.3.53 <code>fgemm()</code> [13/23]	99
15.1.3.54 <code>fgemm()</code> [14/23]	100
15.1.3.55 <code>fgemm()</code> [15/23]	100
15.1.3.56 <code>fgemm()</code> [16/23]	100
15.1.3.57 <code>fgemm()</code> [17/23]	101
15.1.3.58 <code>fgemm()</code> [18/23]	102
15.1.3.59 <code>fsquare()</code> [1/6]	102
15.1.3.60 <code>fsquare()</code> [2/6]	103
15.1.3.61 <code>fsquare()</code> [3/6]	103
15.1.3.62 <code>fsquare()</code> [4/6]	103
15.1.3.63 <code>fsquare()</code> [5/6]	104
15.1.3.64 <code>fgemv()</code> [1/19]	104
15.1.3.65 <code>fgemv()</code> [2/19]	104
15.1.3.66 <code>fgemv()</code> [3/19]	105
15.1.3.67 <code>fgemv()</code> [4/19]	105
15.1.3.68 <code>fgemv()</code> [5/19]	105
15.1.3.69 <code>fgemv()</code> [6/19]	106
15.1.3.70 <code>fgemv()</code> [7/19]	106
15.1.3.71 <code>fgemv()</code> [8/19]	107
15.1.3.72 <code>fgemv()</code> [9/19]	107
15.1.3.73 <code>fgemv()</code> [10/19]	107
15.1.3.74 <code>fgemv()</code> [11/19]	108
15.1.3.75 <code>fgemv()</code> [12/19]	108
15.1.3.76 <code>fgemv()</code> [13/19]	108
15.1.3.77 <code>fgemv()</code> [14/19]	109
15.1.3.78 <code>fgemv()</code> [15/19]	109
15.1.3.79 <code>fgemv()</code> [16/19]	109
15.1.3.80 <code>fger()</code> [1/12]	110
15.1.3.81 <code>fger()</code> [2/12]	110
15.1.3.82 <code>fger()</code> [3/12]	111
15.1.3.83 <code>fger()</code> [4/12]	111
15.1.3.84 <code>fger()</code> [5/12]	111
15.1.3.85 <code>fger()</code> [6/12]	112
15.1.3.86 <code>fger()</code> [7/12]	112
15.1.3.87 <code>fger()</code> [8/12]	112
15.1.3.88 <code>fger()</code> [9/12]	113

15.1.3.89 fger() [10/12]	113
15.1.3.90 fger() [11/12]	113
15.1.3.91 freduce() [1/11]	114
15.1.3.92 freduce() [2/11]	114
15.1.3.93 freduce_constoverride() [1/2]	114
15.1.3.94 finit() [1/8]	115
15.1.3.95 finit() [2/8]	115
15.1.3.96 freduce() [3/11]	115
15.1.3.97 freduce() [4/11]	116
15.1.3.98 pfreduce()	116
15.1.3.99 freduce() [5/11]	116
15.1.3.100 freduce_constoverride() [2/2]	117
15.1.3.101 finit() [3/8]	117
15.1.3.102 finit() [4/8]	117
15.1.3.103 freduce() [6/11]	118
15.1.3.104 freduce() [7/11]	118
15.1.3.105 freivalds()	118
15.1.3.106 fscal() [1/10]	119
15.1.3.107 fscal() [1/10]	119
15.1.3.108 fscal() [2/10]	120
15.1.3.109 fscal() [3/10]	120
15.1.3.110 fscal() [2/10]	121
15.1.3.111 fscal() [3/10]	121
15.1.3.112 fscal() [4/10]	121
15.1.3.113 fscal() [4/10]	121
15.1.3.114 fscal() [5/10]	122
15.1.3.115 fscal() [5/10]	122
15.1.3.116 fscal() [6/10]	122
15.1.3.117 fscal() [6/10]	123
15.1.3.118 fscal() [7/10]	123
15.1.3.119 fscal() [7/10]	123
15.1.3.120 fscal() [8/10]	123
15.1.3.121 fscal() [8/10]	124
15.1.3.122 fsyr2k()	124
15.1.3.123 fsyrk() [1/16]	125
15.1.3.124 fsyrk() [2/16]	125
15.1.3.125 fsyrk() [3/16]	126
15.1.3.126 fsyrk() [4/16]	126
15.1.3.127 fsyrk() [5/16]	127
15.1.3.128 fsyrk() [6/16]	127
15.1.3.129 fsyrk() [7/16]	127
15.1.3.130 fsyrk() [8/16]	128

15.1.3.131 fsyrk() [9/16]	128
15.1.3.132 fsyrk() [10/16]	128
15.1.3.133 fsyrk() [11/16]	129
15.1.3.134 fsyrk() [12/16]	129
15.1.3.135 fsyrk() [13/16]	130
15.1.3.136 fsyrk() [14/16]	130
15.1.3.137 computeS1S2()	132
15.1.3.138 fsyrk() [15/16]	132
15.1.3.139 fsyrk() [16/16]	133
15.1.3.140 fsyrk_strassen() [1/2]	133
15.1.3.141 ftrmm() [1/3]	134
15.1.3.142 ftrmm() [2/3]	134
15.1.3.143 ftrsm() [1/9]	135
15.1.3.144 ftrsm() [2/9]	135
15.1.3.145 ftrsm() [3/9]	136
15.1.3.146 ftrsm() [4/9]	136
15.1.3.147 ftrsm() [5/9]	137
15.1.3.148 cblas_imptrsm()	137
15.1.3.149 ftrsv() [1/2]	137
15.1.3.150 igemm_()	138
15.1.3.151 finit() [5/8]	138
15.1.3.152 fconvert() [1/3]	139
15.1.3.153 fnegin() [1/4]	139
15.1.3.154 fneg() [1/4]	140
15.1.3.155 fzero() [1/5]	140
15.1.3.156 frand() [1/2]	140
15.1.3.157 fiszero() [1/4]	141
15.1.3.158 fequal() [1/4]	141
15.1.3.159 faxpby() [1/2]	142
15.1.3.160 fdot() [9/11]	142
15.1.3.161 fswap() [1/2]	143
15.1.3.162 fzero() [2/5]	143
15.1.3.163 fzero() [3/5]	144
15.1.3.164 frand() [2/2]	144
15.1.3.165 fequal() [2/4]	145
15.1.3.166 fiszero() [2/4]	145
15.1.3.167 fidentity() [1/4]	145
15.1.3.168 fidentity() [2/4]	146
15.1.3.169 finit() [6/8]	146
15.1.3.170 fconvert() [2/3]	146
15.1.3.171 fnegin() [2/4]	147
15.1.3.172 fneg() [2/4]	147

15.1.3.173 faxpby() [2/2]	148
15.1.3.174 fmove() [1/2]	149
15.1.3.175 bitsize()	149
15.1.3.176 bitsize< Givaro::ZRing< Givaro::Integer > >()	149
15.1.3.177 ftrmv()	150
15.1.3.178 ftrsm() [6/9]	150
15.1.3.179 fsyrk_strassen() [2/2]	151
15.1.3.180 pfgemm() [1/7]	151
15.1.3.181 pfgemm_1D_rec()	152
15.1.3.182 pfgemm_2D_rec()	152
15.1.3.183 pfgemm_3D_rec()	153
15.1.3.184 pfgemm_3D_rec2()	153
15.1.3.185 fgemm() [19/23]	153
15.1.3.186 ftrsm() [7/9]	154
15.1.3.187 ftrsm() [8/9]	154
15.1.3.188 sparse_delete() [1/12]	155
15.1.3.189 sparse_delete() [2/12]	155
15.1.3.190 sparse_init() [1/16]	155
15.1.3.191 sparse_init() [2/16]	155
15.1.3.192 sparse_delete() [3/12]	155
15.1.3.193 sparse_delete() [4/12]	156
15.1.3.194 sparse_print() [1/3]	156
15.1.3.195 sparse_init() [3/16]	156
15.1.3.196 sparse_init() [4/16]	156
15.1.3.197 sparse_init() [5/16]	156
15.1.3.198 sparse_init() [6/16]	157
15.1.3.199 sparse_init() [7/16]	157
15.1.3.200 sparse_init() [8/16]	157
15.1.3.201 sparse_delete() [5/12]	157
15.1.3.202 sparse_init() [9/16]	158
15.1.3.203 sparse_delete() [6/12]	158
15.1.3.204 sparse_delete() [7/12]	158
15.1.3.205 sparse_init() [10/16]	158
15.1.3.206 sparse_init() [11/16]	158
15.1.3.207 sparse_delete() [8/12]	159
15.1.3.208 sparse_delete() [9/12]	159
15.1.3.209 sparse_print() [2/3]	159
15.1.3.210 sparse_init() [12/16]	159
15.1.3.211 sparse_init() [13/16]	159
15.1.3.212 sparse_delete() [10/12]	160
15.1.3.213 sparse_init() [14/16]	160
15.1.3.214 operator<<()	160

15.1.3.215 readSmsFormat()	160
15.1.3.216 readSprFormat()	160
15.1.3.217 getDataTypes() [1/4]	161
15.1.3.218 getDataTypes() [2/4]	161
15.1.3.219 getDataTypes() [3/4]	161
15.1.3.220 getDataTypes() [4/4]	161
15.1.3.221 readMachineType()	161
15.1.3.222 readDnsFormat()	161
15.1.3.223 writeDnsFormat()	162
15.1.3.224 fspmv() [1/2]	162
15.1.3.225 sparse_delete() [11/12]	162
15.1.3.226 sparse_delete() [12/12]	162
15.1.3.227 sparse_print() [3/3]	162
15.1.3.228 sparse_init() [15/16]	163
15.1.3.229 sparse_init() [16/16]	163
15.1.3.230 computeDeviation()	163
15.1.3.231 getStat()	163
15.1.3.232 fspmv() [2/2]	164
15.1.3.233 fspmm()	164
15.1.3.234 maxCardinality()	164
15.1.3.235 maxCardinality< Givaro::Modular< int64_t > >()	164
15.1.3.236 maxCardinality< Givaro::Modular< int32_t > >()	164
15.1.3.237 minCardinality()	164
15.1.3.238 fflas_delete() [1/3]	165
15.1.3.239 fflas_delete() [2/3]	165
15.1.3.240 fflas_new() [1/7]	165
15.1.3.241 fflas_new() [2/7]	165
15.1.3.242 finit_rns() [1/2]	165
15.1.3.243 finit_trans_rns()	166
15.1.3.244 fconvert_rns() [1/2]	166
15.1.3.245 fconvert_trans_rns()	166
15.1.3.246 fflas_new() [3/7]	166
15.1.3.247 fflas_new() [4/7]	167
15.1.3.248 finit_rns() [2/2]	167
15.1.3.249 fconvert_rns() [2/2]	167
15.1.3.250 freduce() [8/11]	167
15.1.3.251 freduce() [9/11]	168
15.1.3.252 finit() [7/8]	168
15.1.3.253 fconvert() [3/3]	169
15.1.3.254 fnegin() [3/4]	169
15.1.3.255 fneg() [3/4]	170
15.1.3.256 fzero() [4/5]	170

---

15.1.3.257 fiszero()	[3/4]	170
15.1.3.258 fequal()	[3/4]	171
15.1.3.259 fassign()	[9/10]	171
15.1.3.260 fscaln()	[9/10]	172
15.1.3.261 fscal()	[9/10]	172
15.1.3.262 faxpy()	[5/6]	173
15.1.3.263 fdot()	[10/11]	173
15.1.3.264 fswap()	[2/2]	174
15.1.3.265 fadd()	[5/8]	175
15.1.3.266 fsub()	[3/4]	175
15.1.3.267 faddn()	[4/5]	175
15.1.3.268 fadd()	[6/8]	176
15.1.3.269 fassign()	[10/10]	176
15.1.3.270 fzero()	[5/5]	176
15.1.3.271 fequal()	[4/4]	177
15.1.3.272 fiszero()	[4/4]	177
15.1.3.273 fidentity()	[3/4]	178
15.1.3.274 fidentity()	[4/4]	178
15.1.3.275 freduce()	[10/11]	178
15.1.3.276 freduce()	[11/11]	179
15.1.3.277 finit()	[8/8]	179
15.1.3.278 fnegin()	[4/4]	180
15.1.3.279 fneg()	[4/4]	180
15.1.3.280 fscaln()	[10/10]	180
15.1.3.281 fscal()	[10/10]	181
15.1.3.282 faxpy()	[6/6]	181
15.1.3.283 fmove()	[2/2]	182
15.1.3.284 fadd()	[7/8]	183
15.1.3.285 fsub()	[4/4]	183
15.1.3.286 fsubin()	[3/3]	184
15.1.3.287 fadd()	[8/8]	184
15.1.3.288 faddn()	[5/5]	185
15.1.3.289 fgemv()	[17/19]	185
15.1.3.290 fger()	[12/12]	186
15.1.3.291 ftrsv()	[2/2]	187
15.1.3.292 ftrsm()	[9/9]	187
15.1.3.293 ftrmm()	[3/3]	188
15.1.3.294 fgemm()	[20/23]	189
15.1.3.295 fgemm()	[21/23]	189
15.1.3.296 fgemm()	[22/23]	190
15.1.3.297 fgemm()	[23/23]	190
15.1.3.298 fsquare()	[6/6]	191

15.1.3.299 BlockCuts() [1/2]	191
15.1.3.300 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()	192
15.1.3.301 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()	192
15.1.3.302 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()	192
15.1.3.303 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()	192
15.1.3.304 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()	192
15.1.3.305 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()	193
15.1.3.306 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()	193
15.1.3.307 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()	193
15.1.3.308 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()	193
15.1.3.309 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()	193
15.1.3.310 BlockCuts() [2/2]	194
15.1.3.311 pfzero()	194
15.1.3.312 pfrand()	194
15.1.3.313 fdot() [11/11]	194
15.1.3.314 pfgemm() [2/7]	195
15.1.3.315 pfgemm() [3/7]	195
15.1.3.316 pfgemm() [4/7]	195
15.1.3.317 pfgemm() [5/7]	196
15.1.3.318 pfgemm() [6/7]	196
15.1.3.319 pfgemm() [7/7]	197
15.1.3.320 fgemv() [18/19]	197
15.1.3.321 fgemv() [19/19]	197
15.1.3.322 parseArguments()	198
15.1.3.323 getArgumentValue()	198
15.1.3.324 writeCommandString()	198
15.1.3.325 WriteMatrix() [1/2]	199
15.1.3.326 preamble()	199
15.1.3.327 ReadMatrix() [1/2]	199
15.1.3.328 ReadMatrix() [2/2]	200
15.1.3.329 WriteMatrix() [2/2]	200
15.1.3.330 WritePermutation()	201
15.1.3.331 alignable()	201
15.1.3.332 alignable< Givaro::Integer * >()	201
15.1.3.333 fflas_new() [5/7]	201
15.1.3.334 fflas_new() [6/7]	201
15.1.3.335 fflas_new() [7/7]	202
15.1.3.336 fflas_delete() [3/3]	202
15.1.3.337 prefetch()	202
15.1.3.338 getTLBSize()	202
15.1.3.339 queryCacheSizes()	202
15.1.3.340 queryL1CacheSize()	202

15.1.3.341 queryTopLevelCacheSize()	203
15.1.3.342 getSeed()	203
15.2 FFLAS::ftranspose_impl Namespace Reference	203
15.2.1 Function Documentation	203
15.2.1.1 not_inplace()	203
15.2.1.2 square_inplace()	204
15.2.1.3 nonsquare_inplace_v1()	204
15.2.1.4 nonsquare_inplace_v2()	204
15.3 FFLAS::BLAS3 Namespace Reference	204
15.3.1 Function Documentation	206
15.3.1.1 Bini()	206
15.3.1.2 WinoPar()	206
15.3.1.3 Winograd()	207
15.3.1.4 WinogradAcc_3_23()	207
15.3.1.5 WinogradAcc_3_21()	207
15.3.1.6 WinogradAcc_2_24()	208
15.3.1.7 WinogradAcc_2_27()	208
15.3.1.8 WinogradAcc_LR()	209
15.3.1.9 WinogradAcc_R_S()	209
15.3.1.10 WinogradAcc_L_S()	209
15.3.1.11 Winograd_LR_S()	210
15.3.1.12 Winograd_L_S()	210
15.3.1.13 Winograd_R_S()	211
15.4 FFLAS::csr_hyb_details Namespace Reference	211
15.5 FFLAS::CuttingStrategy Namespace Reference	211
15.5.1 Typedef Documentation	211
15.5.1.1 RNSModulus	211
15.6 FFLAS::details Namespace Reference	212
15.6.1 Function Documentation	213
15.6.1.1 fadd() [1/5]	213
15.6.1.2 fadd() [2/5]	214
15.6.1.3 fadd() [3/5]	214
15.6.1.4 fadd() [4/5]	214
15.6.1.5 fadd() [5/5]	215
15.6.1.6 faxpy() [1/2]	215
15.6.1.7 faxpy() [2/2]	215
15.6.1.8 freduce() [1/4]	215
15.6.1.9 freduce() [2/4]	216
15.6.1.10 freduce() [3/4]	216
15.6.1.11 freduce() [4/4]	216
15.6.1.12 fscaln() [1/2]	216
15.6.1.13 fscal() [1/2]	217

15.6.1.14 fscaln() [2/2]	217
15.6.1.15 fscal() [2/2]	217
15.6.1.16 igebb44()	217
15.6.1.17 igebb24()	218
15.6.1.18 igebb14()	218
15.6.1.19 igebb41()	218
15.6.1.20 igebb21()	219
15.6.1.21 igebb11()	219
15.6.1.22 igebp()	219
15.6.1.23 pack_lhs()	220
15.6.1.24 pack_rhs()	220
15.6.1.25 gebp()	220
15.6.1.26 BlockingFactor()	221
15.7 FFLAS::details_spmv Namespace Reference	221
15.8 FFLAS::ElementCategories Namespace Reference	221
15.9 FFLAS::FieldCategories Namespace Reference	221
15.9.1 Detailed Description	222
15.10 FFLAS::MMHelperAlgo Namespace Reference	222
15.11 FFLAS::ModeCategories Namespace Reference	222
15.11.1 Detailed Description	222
15.12 FFLAS::ParSeqHelper Namespace Reference	222
15.12.1 Detailed Description	223
15.13 FFLAS::Protected Namespace Reference	223
15.13.1 Function Documentation	226
15.13.1.1 computeFactorClassic() [1/3]	226
15.13.1.2 computeFactorClassic() [2/3]	226
15.13.1.3 computeFactorClassic() [3/3]	227
15.13.1.4 DotProdBoundClassic()	227
15.13.1.5 TRSMBound() [1/3]	227
15.13.1.6 TRSMBound() [2/3]	227
15.13.1.7 TRSMBound() [3/3]	227
15.13.1.8 WinogradThreshold() [1/4]	228
15.13.1.9 WinogradThreshold() [2/4]	228
15.13.1.10 WinogradThreshold() [3/4]	228
15.13.1.11 WinogradThreshold() [4/4]	228
15.13.1.12 WinogradSteps()	228
15.13.1.13 DynamicPeeling()	229
15.13.1.14 DynamicPeeling2()	229
15.13.1.15 WinogradCalc()	230
15.13.1.16 fgemm_convert()	230
15.13.1.17 NeedPreAddReduction() [1/2]	230
15.13.1.18 NeedPreAddReduction() [2/2]	231

15.13.1.19 NeedPreSubReduction() [1/2]	231
15.13.1.20 NeedPreSubReduction() [2/2]	231
15.13.1.21 NeedDoublePreAddReduction() [1/2]	231
15.13.1.22 NeedDoublePreAddReduction() [2/2]	232
15.13.1.23 ScalAndReduce() [1/3]	232
15.13.1.24 ScalAndReduce() [2/3]	232
15.13.1.25 fsquareCommon()	232
15.13.1.26 fgemv_convert()	233
15.13.1.27 fger_convert()	233
15.13.1.28 fsyrk_convert()	233
15.13.1.29 ScalAndReduce() [3/3]	234
15.13.1.30 NeedPreScalReduction() [1/2]	234
15.13.1.31 NeedPreScalReduction() [2/2]	234
15.13.1.32 NeedPreAxyReduction() [1/2]	234
15.13.1.33 NeedPreAxyReduction() [2/2]	235
15.13.1.34 min_types() [1/7]	235
15.13.1.35 min_types() [2/7]	235
15.13.1.36 min_types() [3/7]	235
15.13.1.37 min_types() [4/7]	235
15.13.1.38 min_types() [5/7]	235
15.13.1.39 min_types() [6/7]	236
15.13.1.40 min_types() [7/7]	236
15.13.1.41 unfit() [1/4]	236
15.13.1.42 unfit() [2/4]	236
15.13.1.43 unfit() [3/4]	236
15.13.1.44 unfit() [4/4]	236
15.13.1.45 igemm_colmajor() [1/2]	237
15.13.1.46 igemm_colmajor() [2/2]	237
15.13.1.47 igemm()	237
15.13.1.48 MatF2MatD_Triangular()	238
15.13.1.49 MatF2MatFI_Triangular()	238
15.14 FFLAS::sell_details Namespace Reference	238
15.15 FFLAS::sparse_details Namespace Reference	238
15.15.1 Function Documentation	241
15.15.1.1 init_y() [1/2]	241
15.15.1.2 init_y() [2/2]	242
15.15.1.3 fspmv_dispatch() [1/2]	242
15.15.1.4 fspmv_dispatch() [2/2]	242
15.15.1.5 fspmv() [1/12]	242
15.15.1.6 fspmv() [2/12]	243
15.15.1.7 fspmv() [3/12]	243
15.15.1.8 fspmv() [4/12]	243

15.15.1.9 fspmv()	[5/12]	243
15.15.1.10 fspmv()	[6/12]	244
15.15.1.11 fspmv()	[7/12]	244
15.15.1.12 fspmv()	[8/12]	244
15.15.1.13 fspmv()	[9/12]	244
15.15.1.14 fspmm_dispatch()	[1/2]	245
15.15.1.15 fspmm_dispatch()	[2/2]	245
15.15.1.16 fspmm()	[1/9]	245
15.15.1.17 fspmm()	[2/9]	246
15.15.1.18 fspmm()	[3/9]	246
15.15.1.19 fspmm()	[4/9]	246
15.15.1.20 fspmm()	[5/9]	247
15.15.1.21 fspmm()	[6/9]	247
15.15.1.22 fspmm()	[7/9]	247
15.15.1.23 fspmm()	[8/9]	248
15.15.1.24 fspmm()	[9/9]	248
15.15.1.25 pfspmm_dispatch()	[1/2]	248
15.15.1.26 pfspmm_dispatch()	[2/2]	249
15.15.1.27 pfspmm()	[1/9]	249
15.15.1.28 pfspmm()	[2/9]	249
15.15.1.29 pfspmm()	[3/9]	250
15.15.1.30 pfspmm()	[4/9]	250
15.15.1.31 pfspmm()	[5/9]	250
15.15.1.32 pfspmm()	[6/9]	251
15.15.1.33 pfspmm()	[7/9]	251
15.15.1.34 pfspmm()	[8/9]	251
15.15.1.35 pfspmm()	[9/9]	252
15.15.1.36 pfspmv()	[1/6]	252
15.15.1.37 pfspmv()	[2/6]	252
15.15.1.38 pfspmv()	[3/6]	252
15.15.1.39 pfspmv()	[4/6]	253
15.15.1.40 pfspmv()	[5/6]	253
15.15.1.41 pfspmv()	[6/6]	253
15.15.1.42 fspmv()	[10/12]	253
15.15.1.43 fspmv()	[11/12]	254
15.15.1.44 fspmv()	[12/12]	254
15.16 FFLAS::sparse_details_impl Namespace Reference		254
15.16.1 Function Documentation		262
15.16.1.1 fspmm()	[1/15]	263
15.16.1.2 fspmm()	[2/15]	263
15.16.1.3 fspmm()	[3/15]	263
15.16.1.4 fspmm_simd_aligned()	[1/2]	263

15.16.1.5 fspmm_simd_unaligned() [1/2]	264
15.16.1.6 fspmm_one() [1/4]	264
15.16.1.7 fspmm_mone() [1/4]	264
15.16.1.8 fspmm_one_simd_aligned() [1/3]	264
15.16.1.9 fspmm_one_simd_unaligned() [1/3]	265
15.16.1.10 fspmm_mone_simd_aligned() [1/3]	265
15.16.1.11 fspmm_mone_simd_unaligned() [1/3]	265
15.16.1.12 fspmv() [1/21]	265
15.16.1.13 fspmv() [2/21]	266
15.16.1.14 fspmv() [3/21]	266
15.16.1.15 fspmv_one() [1/10]	266
15.16.1.16 fspmv_mone() [1/10]	266
15.16.1.17 fspmv_one() [2/10]	266
15.16.1.18 fspmv_mone() [2/10]	267
15.16.1.19 pfspmm() [1/18]	267
15.16.1.20 pfspmm() [2/18]	267
15.16.1.21 pfspmm() [3/18]	267
15.16.1.22 pfspmm_one() [1/2]	268
15.16.1.23 pfspmm_mone() [1/2]	268
15.16.1.24 pfspmm_one() [2/2]	268
15.16.1.25 pfspmm_mone() [2/2]	268
15.16.1.26 pfspmv() [1/18]	269
15.16.1.27 pfspmv_task()	269
15.16.1.28 pfspmv() [2/18]	269
15.16.1.29 pfspmv() [3/18]	269
15.16.1.30 pfspmv_one() [1/8]	269
15.16.1.31 pfspmv_mone() [1/8]	270
15.16.1.32 pfspmv_one() [2/8]	270
15.16.1.33 pfspmv_mone() [2/8]	270
15.16.1.34 fspmm() [4/15]	270
15.16.1.35 fspmm() [5/15]	271
15.16.1.36 fspmm_simd_aligned() [2/2]	271
15.16.1.37 fspmm_simd_unaligned() [2/2]	271
15.16.1.38 fspmm() [6/15]	271
15.16.1.39 fspmm_one() [2/4]	272
15.16.1.40 fspmm_mone() [2/4]	272
15.16.1.41 fspmm_one_simd_aligned() [2/3]	272
15.16.1.42 fspmm_one_simd_unaligned() [2/3]	272
15.16.1.43 fspmm_mone_simd_aligned() [2/3]	273
15.16.1.44 fspmm_mone_simd_unaligned() [2/3]	273
15.16.1.45 fspmv() [4/21]	273
15.16.1.46 fspmv() [5/21]	273

15.16.1.47 fspmv()	[ 6/21]	274
15.16.1.48 fspmv_one()	[ 3/10]	274
15.16.1.49 fspmv_mone()	[ 3/10]	274
15.16.1.50 fspmv_one()	[ 4/10]	274
15.16.1.51 fspmv_mone()	[ 4/10]	274
15.16.1.52 pfspmm()	[ 4/18]	275
15.16.1.53 pfspmm()	[ 5/18]	275
15.16.1.54 pfspmm()	[ 6/18]	275
15.16.1.55 pfspmm()	[ 7/18]	275
15.16.1.56 pfspmm()	[ 8/18]	276
15.16.1.57 pfspmm()	[ 9/18]	276
15.16.1.58 pfspmv()	[ 4/18]	276
15.16.1.59 pfspmv()	[ 5/18]	276
15.16.1.60 pfspmv()	[ 6/18]	277
15.16.1.61 fspmm()	[ 7/15]	277
15.16.1.62 fspmm()	[ 8/15]	277
15.16.1.63 fspmm()	[ 9/15]	277
15.16.1.64 fspmv()	[ 7/21]	278
15.16.1.65 fspmv()	[ 8/21]	278
15.16.1.66 fspmv()	[ 9/21]	278
15.16.1.67 pfspmm()	[10/18]	278
15.16.1.68 pfspmm()	[11/18]	278
15.16.1.69 pfspmm()	[12/18]	279
15.16.1.70 pfspmm()	[13/18]	279
15.16.1.71 pfspmm()	[14/18]	279
15.16.1.72 pfspmm()	[15/18]	279
15.16.1.73 pfspmm_zo()	[ 1/2]	280
15.16.1.74 pfspmm_zo()	[ 2/2]	280
15.16.1.75 pfspmv()	[ 7/18]	280
15.16.1.76 pfspmv()	[ 8/18]	280
15.16.1.77 pfspmv()	[ 9/18]	281
15.16.1.78 pfspmv_one()	[ 3/8]	281
15.16.1.79 pfspmv_mone()	[ 3/8]	281
15.16.1.80 pfspmv_one()	[ 4/8]	281
15.16.1.81 pfspmv_mone()	[ 4/8]	281
15.16.1.82 fspmm()	[10/15]	282
15.16.1.83 fspmm()	[11/15]	282
15.16.1.84 fspmm()	[12/15]	282
15.16.1.85 fspmm_mone()	[ 3/4]	282
15.16.1.86 fspmm_one()	[ 3/4]	283
15.16.1.87 fspmm_mone()	[ 4/4]	283
15.16.1.88 fspmm_one()	[ 4/4]	283

15.16.1.89 fspmm_one_simd_aligned() [3/3]	283
15.16.1.90 fspmm_one_simd_unaligned() [3/3]	284
15.16.1.91 fspmm_mone_simd_aligned() [3/3]	284
15.16.1.92 fspmm_mone_simd_unaligned() [3/3]	284
15.16.1.93 fspmv() [10/21]	284
15.16.1.94 fspmv() [11/21]	285
15.16.1.95 fspmv() [12/21]	285
15.16.1.96 fspmv_one() [5/10]	285
15.16.1.97 fspmv_mone() [5/10]	285
15.16.1.98 fspmv_one() [6/10]	285
15.16.1.99 fspmv_mone() [6/10]	286
15.16.1.100 pfspmv() [10/18]	286
15.16.1.101 pfspmv() [11/18]	286
15.16.1.102 pfspmv() [12/18]	286
15.16.1.103 pfspmv_one() [5/8]	286
15.16.1.104 pfspmv_mone() [5/8]	287
15.16.1.105 pfspmv_one() [6/8]	287
15.16.1.106 pfspmv_mone() [6/8]	287
15.16.1.107 fspmv() [13/21]	287
15.16.1.108 fspmv_simd() [1/4]	287
15.16.1.109 fspmv() [14/21]	288
15.16.1.110 fspmv_simd() [2/4]	288
15.16.1.111 fspmv() [15/21]	288
15.16.1.112 fspmv_one() [7/10]	288
15.16.1.113 fspmv_mone() [7/10]	288
15.16.1.114 fspmv_one() [8/10]	289
15.16.1.115 fspmv_mone() [8/10]	289
15.16.1.116 fspmv_one_simd() [1/2]	289
15.16.1.117 fspmv_mone_simd() [1/2]	289
15.16.1.118 pfspmm() [16/18]	289
15.16.1.119 pfspmm() [17/18]	290
15.16.1.120 pfspmm() [18/18]	290
15.16.1.121 pfspmv() [13/18]	290
15.16.1.122 pfspmv() [14/18]	290
15.16.1.123 pfspmv() [15/18]	291
15.16.1.124 fspmm() [13/15]	291
15.16.1.125 fspmm() [14/15]	291
15.16.1.126 fspmm() [15/15]	291
15.16.1.127 fspmv() [16/21]	292
15.16.1.128 fspmv() [17/21]	292
15.16.1.129 fspmv() [18/21]	292
15.16.1.130 pfspmv() [16/18]	292

15.16.1.131 pfspmv()	[17/18]	292
15.16.1.132 pfspmv()	[18/18]	293
15.16.1.133 pfspmv_one()	[7/8]	293
15.16.1.134 pfspmv_mone()	[7/8]	293
15.16.1.135 pfspmv_one()	[8/8]	293
15.16.1.136 pfspmv_mone()	[8/8]	293
15.16.1.137 fspmv()	[19/21]	294
15.16.1.138 fspmv_simd()	[3/4]	294
15.16.1.139 fspmv()	[20/21]	294
15.16.1.140 fspmv_simd()	[4/4]	294
15.16.1.141 fspmv()	[21/21]	294
15.16.1.142 fspmv_one()	[9/10]	295
15.16.1.143 fspmv_mone()	[9/10]	295
15.16.1.144 fspmv_one_simd()	[2/2]	295
15.16.1.145 fspmv_mone_simd()	[2/2]	295
15.16.1.146 fspmv_one()	[10/10]	295
15.16.1.147 fspmv_mone()	[10/10]	296
15.17 FFLAS::StrategyParameter Namespace Reference		296
15.18 FFLAS::StructureHelper Namespace Reference		296
15.18.1 Detailed Description		296
15.19 FFLAS::vectorised Namespace Reference		296
15.19.1 Function Documentation		298
15.19.1.1 VEC_ADD()		298
15.19.1.2 addp()		298
15.19.1.3 VEC_SUB()		299
15.19.1.4 subp()		299
15.19.1.5 add()		299
15.19.1.6 sub()		299
15.19.1.7 axpyp()	[1/2]	300
15.19.1.8 axpyp()	[2/2]	300
15.19.1.9 reduce()	[1/9]	300
15.19.1.10 reduce()	[2/9]	300
15.19.1.11 reduce()	[3/9]	300
15.19.1.12 reduce()	[4/9]	301
15.19.1.13 reduce()	[5/9]	301
15.19.1.14 reduce()	[6/9]	301
15.19.1.15 reduce()	[7/9]	301
15.19.1.16 reduce()	[8/9]	301
15.19.1.17 reduce()	[9/9]	302
15.19.1.18 modp()	[1/2]	302
15.19.1.19 modp()	[2/2]	302
15.19.1.20 scalp()	[1/3]	302

15.19.1.21 scalp() [2/3]	302
15.19.1.22 scalp() [3/3]	303
15.20 FFLAS::vectorised::unswitch Namespace Reference	303
15.20.1 Function Documentation	303
15.20.1.1 axpyp() [1/2]	304
15.20.1.2 axpyp() [2/2]	304
15.20.1.3 modp() [1/2]	304
15.20.1.4 modp() [2/2]	304
15.20.1.5 scalp() [1/3]	305
15.20.1.6 scalp() [2/3]	305
15.20.1.7 scalp() [3/3]	305
15.21 FFPACK Namespace Reference	305
15.21.1 Detailed Description	322
15.21.2 Typedef Documentation	322
15.21.2.1 Checker_PLUQ	322
15.21.2.2 Checker_Det	322
15.21.2.3 Checker_invert	322
15.21.2.4 Checker_charpoly	322
15.21.2.5 ForceCheck_PLUQ	322
15.21.2.6 ForceCheck_Det	323
15.21.2.7 ForceCheck_invert	323
15.21.2.8 ForceCheck_charpoly	323
15.21.3 Function Documentation	323
15.21.3.1 LAPACKPerm2MathPerm()	323
15.21.3.2 MathPerm2LAPACKPerm()	323
15.21.3.3 applyP() [1/4]	323
15.21.3.4 applyP() [2/4]	324
15.21.3.5 applyP() [3/4]	325
15.21.3.6 MonotonicApplyP()	325
15.21.3.7 fgetrs() [1/4]	326
15.21.3.8 fgetrs() [2/4]	326
15.21.3.9 fgesv() [1/4]	327
15.21.3.10 fgesv() [2/4]	328
15.21.3.11 ftrtri() [1/2]	329
15.21.3.12 trinv_left() [1/2]	329
15.21.3.13 ftrtrm() [1/2]	329
15.21.3.14 ftrstr()	330
15.21.3.15 ftrssyr2k()	331
15.21.3.16 fsytrf() [1/3]	331
15.21.3.17 fsytrf() [2/3]	332
15.21.3.18 fsytrf() [3/3]	332
15.21.3.19 fsytrf_nonunit() [1/3]	332

15.21.3.20 PLUQ()	[1/6]	333
15.21.3.21 pPLUQ()		333
15.21.3.22 PLUQ()	[2/6]	334
15.21.3.23 PLUQ()	[3/6]	334
15.21.3.24 LUdivine()	[1/4]	334
15.21.3.25 ColumnEchelonForm()	[1/3]	335
15.21.3.26 pColumnEchelonForm()		336
15.21.3.27 ColumnEchelonForm()	[2/3]	336
15.21.3.28 RowEchelonForm()	[1/3]	336
15.21.3.29 pRowEchelonForm()		337
15.21.3.30 RowEchelonForm()	[2/3]	337
15.21.3.31 ReducedColumnEchelonForm()	[1/3]	338
15.21.3.32 pReducedColumnEchelonForm()		338
15.21.3.33 ReducedColumnEchelonForm()	[2/3]	339
15.21.3.34 ReducedRowEchelonForm()	[1/3]	339
15.21.3.35 pReducedRowEchelonForm()		339
15.21.3.36 ReducedRowEchelonForm()	[2/3]	340
15.21.3.37 Invert()	[1/4]	340
15.21.3.38 Invert()	[2/4]	341
15.21.3.39 Invert2()	[1/2]	341
15.21.3.40 CharPoly()	[1/8]	342
15.21.3.41 CharPoly()	[2/8]	343
15.21.3.42 CharPoly()	[3/8]	343
15.21.3.43 MinPoly()	[1/4]	344
15.21.3.44 MinPoly()	[2/4]	344
15.21.3.45 MatVecMinPoly()	[1/2]	345
15.21.3.46 Rank()	[1/3]	345
15.21.3.47 pRank()		346
15.21.3.48 Rank()	[2/3]	346
15.21.3.49 IsSingular()	[1/2]	346
15.21.3.50 Det()	[1/6]	347
15.21.3.51 pDet()		347
15.21.3.52 Det()	[2/6]	348
15.21.3.53 Solve()	[1/3]	348
15.21.3.54 Solve()	[2/3]	348
15.21.3.55 pSolve()		349
15.21.3.56 RandomNullSpaceVector()	[1/3]	349
15.21.3.57 NullSpaceBasis()	[1/2]	350
15.21.3.58 RowRankProfile()	[1/3]	350
15.21.3.59 pRowRankProfile()		351
15.21.3.60 RowRankProfile()	[2/3]	351
15.21.3.61 ColumnRankProfile()	[1/3]	351

15.21.3.62 pColumnRankProfile()	352
15.21.3.63 ColumnRankProfile() [2/3]	352
15.21.3.64 RankProfileFromLU()	352
15.21.3.65 LeadingSubmatrixRankProfiles()	353
15.21.3.66 RowRankProfileSubmatrixIndices() [1/2]	354
15.21.3.67 ColRankProfileSubmatrixIndices() [1/2]	354
15.21.3.68 RowRankProfileSubmatrix() [1/2]	355
15.21.3.69 ColRankProfileSubmatrix() [1/2]	356
15.21.3.70 getTriangular() [1/2]	356
15.21.3.71 getTriangular() [2/2]	357
15.21.3.72 getEchelonForm() [1/2]	358
15.21.3.73 getEchelonForm() [2/2]	358
15.21.3.74 getEchelonTransform()	359
15.21.3.75 getReducedEchelonForm() [1/2]	360
15.21.3.76 getReducedEchelonForm() [2/2]	361
15.21.3.77 getReducedEchelonTransform()	361
15.21.3.78 PLUQtoEchelonPermutation()	362
15.21.3.79 LTBruhatGen()	362
15.21.3.80 getLTBruhatGen() [1/2]	363
15.21.3.81 getLTBruhatGen() [2/2]	363
15.21.3.82 LTQSorter()	364
15.21.3.83 CompressToBlockBiDiagonal()	364
15.21.3.84 ExpandBlockBiDiagonalToBruhat()	365
15.21.3.85 Bruhat2EchelonPermutation()	366
15.21.3.86 TInverter() [1/2]	366
15.21.3.87 ComputeRPermutation() [1/2]	367
15.21.3.88 productBruhatxTS() [1/2]	367
15.21.3.89 LQUPtoInverseOfFullRankMinor() [1/2]	367
15.21.3.90 RandomNullSpaceVector() [2/3]	368
15.21.3.91 solveLB() [1/2]	368
15.21.3.92 solveLB2() [1/2]	369
15.21.3.93 TInverter() [2/2]	369
15.21.3.94 ComputeRPermutation() [2/2]	369
15.21.3.95 expandLCRE()	370
15.21.3.96 productBruhatxTS() [2/2]	370
15.21.3.97 Danilevski()	371
15.21.3.98 buildMatrix()	372
15.21.3.99 CharPoly() [4/8]	372
15.21.3.100 CharPoly() [5/8]	372
15.21.3.101 Det() [3/6]	372
15.21.3.102 Det() [4/6]	373
15.21.3.103 fsytrf_BC_Crout()	373

15.21.3.104 fsytrf_BC_RL()	373
15.21.3.105 fsytrf_UP_RPM_BC_RL()	373
15.21.3.106 fsytrf_LOW_RPM_BC_Crout()	374
15.21.3.107 fsytrf_UP_RPM_BC_Crout()	374
15.21.3.108 fsytrf_UP_RPM()	374
15.21.3.109 fsytrf_nonunit() [2/3]	374
15.21.3.110 fsytrf_nonunit() [3/3]	375
15.21.3.111 fsytrf_RPM()	375
15.21.3.112 getTridiagonal()	375
15.21.3.113 LUdivine_gauss() [1/2]	375
15.21.3.114 LUdivine_small() [1/2]	376
15.21.3.115 LUdivine() [2/4]	376
15.21.3.116 LUdivine() [3/4]	376
15.21.3.117 MonotonicCompress()	377
15.21.3.118 MonotonicCompressMorePivots()	377
15.21.3.119 MonotonicCompressCycles()	377
15.21.3.120 MonotonicExpand()	378
15.21.3.121 applyP_block()	378
15.21.3.122 doApplyS()	378
15.21.3.123 MatrixApplyS() [1/3]	379
15.21.3.124 MatrixApplyS() [2/3]	379
15.21.3.125 MatrixApplyS() [3/3]	379
15.21.3.126 PermApplyS()	380
15.21.3.127 doApplyT()	380
15.21.3.128 MatrixApplyT() [1/3]	380
15.21.3.129 MatrixApplyT() [2/3]	380
15.21.3.130 MatrixApplyT() [3/3]	381
15.21.3.131 PermApplyT()	381
15.21.3.132 composePermutationsLLL()	381
15.21.3.133 composePermutationsLLM()	382
15.21.3.134 composePermutationsMLM()	382
15.21.3.135 cyclic_shift_mathPerm()	382
15.21.3.136 cyclic_shift_row_col() [1/2]	383
15.21.3.137 cyclic_shift_row() [1/3]	383
15.21.3.138 cyclic_shift_row() [2/3]	383
15.21.3.139 cyclic_shift_col() [1/3]	383
15.21.3.140 cyclic_shift_col() [2/3]	383
15.21.3.141 PLUQ_basecaseV3()	384
15.21.3.142 PLUQ_basecaseV2()	384
15.21.3.143 PLUQ_basecaseCrout()	384
15.21.3.144 _PLUQ()	384
15.21.3.145 PLUQ() [4/6]	385

15.21.3.146 threads_fgemm()	385
15.21.3.147 threads_ftrsm()	385
15.21.3.148 PLUQ() [5/6]	385
15.21.3.149 fflas_const_cast() [1/3]	386
15.21.3.150 fflas_const_cast() [2/3]	386
15.21.3.151 cyclic_shift_row_col() [2/2]	386
15.21.3.152 cyclic_shift_row() [3/3]	386
15.21.3.153 cyclic_shift_col() [3/3]	386
15.21.3.154 applyP() [4/4]	387
15.21.3.155 fgetrs() [3/4]	387
15.21.3.156 fgetrs() [4/4]	387
15.21.3.157 fgesv() [3/4]	388
15.21.3.158 fgesv() [4/4]	388
15.21.3.159 ftrtri() [2/2]	388
15.21.3.160 trinv_left() [2/2]	388
15.21.3.161 ftrtrm() [2/2]	389
15.21.3.162 PLUQ() [6/6]	389
15.21.3.163 LUdivine() [4/4]	389
15.21.3.164 LUdivine_small() [2/2]	389
15.21.3.165 LUdivine_gauss() [2/2]	390
15.21.3.166 RowEchelonForm() [3/3]	390
15.21.3.167 ReducedRowEchelonForm() [3/3]	390
15.21.3.168 ColumnEchelonForm() [3/3]	390
15.21.3.169 ReducedColumnEchelonForm() [3/3]	391
15.21.3.170 Invert() [3/4]	391
15.21.3.171 Invert() [4/4]	391
15.21.3.172 Invert2() [2/2]	391
15.21.3.173 CharPoly() [6/8]	392
15.21.3.174 CharPoly() [7/8]	392
15.21.3.175 CharPoly() [8/8]	392
15.21.3.176 MinPoly() [3/4]	392
15.21.3.177 MinPoly() [4/4]	393
15.21.3.178 MatVecMinPoly() [2/2]	393
15.21.3.179 KrylovElim()	393
15.21.3.180 SpecRankProfile()	393
15.21.3.181 Rank() [3/3]	394
15.21.3.182 IsSingular() [2/2]	394
15.21.3.183 Det() [5/6]	394
15.21.3.184 Det() [6/6]	394
15.21.3.185 Solve() [3/3]	395
15.21.3.186 solveLB() [2/2]	395
15.21.3.187 solveLB2() [2/2]	395

15.21.3.188 RandomNullSpaceVector()	[ 3/3 ]	395
15.21.3.189 NullSpaceBasis()	[ 2/2 ]	396
15.21.3.190 RowRankProfile()	[ 3/3 ]	396
15.21.3.191 ColumnRankProfile()	[ 3/3 ]	396
15.21.3.192 RowRankProfileSubmatrixIndices()	[ 2/2 ]	396
15.21.3.193 ColRankProfileSubmatrixIndices()	[ 2/2 ]	397
15.21.3.194 RowRankProfileSubmatrix()	[ 2/2 ]	397
15.21.3.195 ColRankProfileSubmatrix()	[ 2/2 ]	397
15.21.3.196 getTriangular< FFLAS_FIELD< FFLAS_ELT > >()	[ 1/2 ]	397
15.21.3.197 getTriangular< FFLAS_FIELD< FFLAS_ELT > >()	[ 2/2 ]	398
15.21.3.198 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()	[ 1/2 ]	398
15.21.3.199 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()	[ 2/2 ]	398
15.21.3.200 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()		399
15.21.3.201 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()	[ 1/2 ]	399
15.21.3.202 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()	[ 2/2 ]	399
15.21.3.203 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()		400
15.21.3.204 LQUPtoInverseOfFullRankMinor()	[ 2/2 ]	400
15.21.3.205 fflas_const_cast()	[ 3/3 ]	400
15.21.3.206 failure()		400
15.21.3.207 isOdd()	[ 1/3 ]	400
15.21.3.208 isOdd()	[ 2/3 ]	401
15.21.3.209 isOdd()	[ 3/3 ]	401
15.21.3.210 NonZeroRandomMatrix()	[ 1/2 ]	401
15.21.3.211 NonZeroRandomMatrix()	[ 2/2 ]	401
15.21.3.212 RandomMatrix()	[ 1/2 ]	402
15.21.3.213 RandomMatrix()	[ 2/2 ]	403
15.21.3.214 RandomTriangularMatrix()	[ 1/2 ]	403
15.21.3.215 RandomTriangularMatrix()	[ 2/2 ]	404
15.21.3.216 RandInt()		404
15.21.3.217 RandomSymmetricMatrix()		405
15.21.3.218 RandomMatrixWithRank()	[ 1/2 ]	405
15.21.3.219 RandomMatrixWithRank()	[ 2/2 ]	406
15.21.3.220 RandomIndexSubset()		406
15.21.3.221 RandomPermutation()		407
15.21.3.222 RandomRankProfileMatrix()		407
15.21.3.223 swapval()		407
15.21.3.224 RandomSymmetricRankProfileMatrix()		408
15.21.3.225 RandomLTQSRankProfileMatrix()		408
15.21.3.226 RandomMatrixWithRankandRPM()	[ 1/2 ]	408
15.21.3.227 RandomMatrixWithRankandRPM()	[ 2/2 ]	409
15.21.3.228 RandomSymmetricMatrixWithRankandRPM()	[ 1/2 ]	409
15.21.3.229 RandomSymmetricMatrixWithRankandRPM()	[ 2/2 ]	410

15.21.3.230 RandomMatrixWithRankandRandomRPM() [1/2]	411
15.21.3.231 RandomMatrixWithRankandRandomRPM() [2/2]	411
15.21.3.232 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]	412
15.21.3.233 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]	412
15.21.3.234 RandomMatrixWithDet() [1/2]	414
15.21.3.235 RandomMatrixWithDet() [2/2]	414
15.21.3.236 RandomLTQSMMatrixWithRankandQSorder()	416
15.21.3.237 chooseField()	416
15.21.3.238 chooseField< Givaro::ZRing< int32_t > >()	416
15.21.3.239 chooseField< Givaro::ZRing< int64_t > >()	416
15.21.3.240 chooseField< Givaro::ZRing< float > >()	417
15.21.3.241 chooseField< Givaro::ZRing< double > >()	417
15.22 FFPACK::Protected Namespace Reference	417
15.22.1 Function Documentation	418
15.22.1.1 LUdivine_construct() [1/2]	419
15.22.1.2 GaussJordan()	419
15.22.1.3 KellerGehrig()	420
15.22.1.4 KGFast()	420
15.22.1.5 KGFast_generalized()	420
15.22.1.6 fgemv_kgf()	420
15.22.1.7 LUKrylov()	420
15.22.1.8 Danilevski()	421
15.22.1.9 RandomKrylovPrecond()	421
15.22.1.10 ArithProg()	421
15.22.1.11 LUKrylov_KGFast()	421
15.22.1.12 MatVecMinPoly()	422
15.22.1.13 Hybrid_KGF_LUK_MinPoly()	422
15.22.1.14 updateD()	422
15.22.1.15 newD()	422
15.22.1.16 CompressRows()	423
15.22.1.17 CompressRowsQK()	423
15.22.1.18 DeCompressRows()	423
15.22.1.19 DeCompressRowsQK()	423
15.22.1.20 CompressRowsQA()	423
15.22.1.21 DeCompressRowsQA()	424
15.22.1.22 LUdivine_construct() [2/2]	424
15.23 Givaro Namespace Reference	424
15.24 MKL_CONFIG Namespace Reference	424
15.25 Reclnt Namespace Reference	424
<b>16 Data Structure Documentation</b>	<b>427</b>
16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference	427

16.1.1 Member Typedef Documentation	427
16.1.1.1 value	427
16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference	427
16.2.1 Member Typedef Documentation	427
16.2.1.1 value	427
16.3 ALL< v > Struct Template Reference	427
16.4 ALL< false, v... > Struct Template Reference	428
16.4.1 Field Documentation	428
16.4.1.1 value	428
16.5 ALL< true, v... > Struct Template Reference	428
16.5.1 Field Documentation	428
16.5.1.1 value	428
16.6 ALL<> Struct Reference	428
16.6.1 Field Documentation	428
16.6.1.1 value	428
16.7 ArbitraryPrecIntTag Struct Reference	428
16.7.1 Detailed Description	429
16.8 AreEqual< X, Y > Class Template Reference	429
16.8.1 Field Documentation	429
16.8.1.1 value	429
16.9 AreEqual< X, X > Class Template Reference	429
16.9.1 Field Documentation	429
16.9.1.1 value	429
16.10 Argument Struct Reference	429
16.10.1 Field Documentation	430
16.10.1.1 c	430
16.10.1.2 example	430
16.10.1.3 helpString	430
16.10.1.4 type	430
16.10.1.5 data	430
16.11 associatedDelayedField< Field > Struct Template Reference	430
16.11.1 Member Typedef Documentation	430
16.11.1.1 field	430
16.11.1.2 type	430
16.12 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference	430
16.12.1 Member Typedef Documentation	431
16.12.1.1 field	431
16.12.1.2 type	431
16.13 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference	431
16.13.1 Member Typedef Documentation	431
16.13.1.1 field	431

16.13.1.2 type . . . . .	431
16.14 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference . . . . .	431
16.14.1 Member Typedef Documentation . . . . .	432
16.14.1.1 field . . . . .	432
16.14.1.2 type . . . . .	432
16.15 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference . . . . .	432
16.15.1 Member Typedef Documentation . . . . .	432
16.15.1.1 field . . . . .	432
16.15.1.2 type . . . . .	432
16.16 Auto Struct Reference . . . . .	432
16.17 Bench< Elt > Class Template Reference . . . . .	432
16.17.1 Member Typedef Documentation . . . . .	433
16.17.1.1 Field . . . . .	433
16.17.1.2 Elt_ptr . . . . .	433
16.17.1.3 Residu . . . . .	433
16.17.1.4 enable_if_t . . . . .	433
16.17.1.5 is_same_element . . . . .	434
16.17.1.6 enable_if_no_simd_t . . . . .	434
16.17.1.7 enable_if_simd128_t . . . . .	434
16.17.1.8 enable_if_simd256_t . . . . .	434
16.17.1.9 enable_if_simd512_t . . . . .	434
16.17.2 Constructor & Destructor Documentation . . . . .	434
16.17.2.1 Bench() . . . . .	434
16.17.3 Member Function Documentation . . . . .	434
16.17.3.1 cardinality() [1/2] . . . . .	434
16.17.3.2 cardinality() [2/2] . . . . .	434
16.17.3.3 doBenchs() . . . . .	434
16.17.3.4 run() . . . . .	434
16.17.4 Field Documentation . . . . .	435
16.17.4.1 F . . . . .	435
16.17.4.2 m . . . . .	435
16.17.4.3 n . . . . .	435
16.17.4.4 iters . . . . .	435
16.17.4.5 inplace . . . . .	435
16.18 Bini Struct Reference . . . . .	435
16.19 Block Struct Reference . . . . .	435
16.20 BlockTransposeSIMD< Field, Simd, > Struct Template Reference . . . . .	435
16.20.1 Member Function Documentation . . . . .	436
16.20.1.1 size() . . . . .	436
16.20.1.2 info() . . . . .	436
16.20.1.3 transpose() [1/5] . . . . .	436
16.20.1.4 transpose() [2/5] . . . . .	436

16.20.1.5 transpose() [3/5]	436
16.20.1.6 transpose() [4/5]	436
16.20.1.7 transpose() [5/5]	437
16.21 callLUdivine_small< Element > Class Template Reference	437
16.21.1 Member Function Documentation	437
16.21.1.1 operator()()	437
16.22 callLUdivine_small< double > Class Reference	437
16.22.1 Member Function Documentation	437
16.22.1.1 operator()()	437
16.23 callLUdivine_small< float > Class Reference	438
16.23.1 Member Function Documentation	438
16.23.1.1 operator()()	438
16.24 CharpolyFailed Class Reference	438
16.25 Checker_Empty< Field > Struct Template Reference	438
16.25.1 Constructor & Destructor Documentation	439
16.25.1.1 Checker_Empty()	439
16.25.2 Member Function Documentation	439
16.25.2.1 check()	439
16.26 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference	439
16.26.1 Constructor & Destructor Documentation	439
16.26.1.1 CheckerImplem_charpoly() [1/2]	439
16.26.1.2 CheckerImplem_charpoly() [2/2]	439
16.26.1.3 ~CheckerImplem_charpoly()	440
16.26.2 Member Function Documentation	440
16.26.2.1 check()	440
16.27 CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > Class Template Reference	440
16.27.1 Member Typedef Documentation	440
16.27.1.1 Ring	440
16.27.2 Constructor & Destructor Documentation	440
16.27.2.1 CheckerImplem_charpoly() [1/2]	440
16.27.2.2 CheckerImplem_charpoly() [2/2]	441
16.27.2.3 ~CheckerImplem_charpoly()	441
16.27.3 Member Function Documentation	441
16.27.3.1 check()	441
16.28 CheckerImplem_Det< Field > Class Template Reference	441
16.28.1 Constructor & Destructor Documentation	441
16.28.1.1 CheckerImplem_Det() [1/2]	441
16.28.1.2 CheckerImplem_Det() [2/2]	441
16.28.1.3 ~CheckerImplem_Det()	442
16.28.2 Member Function Documentation	442
16.28.2.1 check()	442

16.29 CheckerImplem_fgemm< Field > Class Template Reference	442
16.29.1 Constructor & Destructor Documentation	442
16.29.1.1 CheckerImplem_fgemm() [1/2]	442
16.29.1.2 CheckerImplem_fgemm() [2/2]	443
16.29.1.3 ~CheckerImplem_fgemm()	443
16.29.2 Member Function Documentation	443
16.29.2.1 check()	443
16.30 CheckerImplem_ftdsm< Field > Class Template Reference	443
16.30.1 Constructor & Destructor Documentation	443
16.30.1.1 CheckerImplem_ftdsm() [1/2]	443
16.30.1.2 CheckerImplem_ftdsm() [2/2]	444
16.30.1.3 ~CheckerImplem_ftdsm()	444
16.30.2 Member Function Documentation	444
16.30.2.1 check()	444
16.31 CheckerImplem_invert< Field > Class Template Reference	444
16.31.1 Constructor & Destructor Documentation	444
16.31.1.1 CheckerImplem_invert() [1/2]	445
16.31.1.2 CheckerImplem_invert() [2/2]	445
16.31.1.3 ~CheckerImplem_invert()	445
16.31.2 Member Function Documentation	445
16.31.2.1 check()	445
16.32 CheckerImplem_PLUQ< Field > Class Template Reference	445
16.32.1 Constructor & Destructor Documentation	445
16.32.1.1 CheckerImplem_PLUQ() [1/2]	445
16.32.1.2 CheckerImplem_PLUQ() [2/2]	446
16.32.1.3 ~CheckerImplem_PLUQ()	446
16.32.2 Member Function Documentation	446
16.32.2.1 check()	446
16.33 Classic Struct Reference	446
16.34 Column Struct Reference	446
16.35 CompactElement< Element > Struct Template Reference	447
16.35.1 Member Typedef Documentation	447
16.35.1.1 type	447
16.36 CompactElement< double > Struct Reference	447
16.36.1 Member Typedef Documentation	447
16.36.1.1 type	447
16.37 CompactElement< float > Struct Reference	447
16.37.1 Member Typedef Documentation	447
16.37.1.1 type	447
16.38 CompactElement< int16_t > Struct Reference	447
16.38.1 Member Typedef Documentation	448
16.38.1.1 type	448

16.39 CompactElement< int32_t > Struct Reference	448
16.39.1 Member Typedef Documentation	448
16.39.1.1 type	448
16.40 CompactElement< int64_t > Struct Reference	448
16.40.1 Member Typedef Documentation	448
16.40.1.1 type	448
16.41 compatible_data_type< Field > Struct Template Reference	448
16.41.1 Field Documentation	448
16.41.1.1 value	448
16.42 compatible_data_type< Givaro::ZRing< double > > Struct Reference	449
16.42.1 Field Documentation	449
16.42.1.1 value	449
16.43 compatible_data_type< Givaro::ZRing< float > > Struct Reference	449
16.43.1 Field Documentation	449
16.43.1.1 value	449
16.44 Compose< H1, H2 > Struct Template Reference	449
16.44.1 Constructor & Destructor Documentation	449
16.44.1.1 Compose() [1/5]	450
16.44.1.2 Compose() [2/5]	450
16.44.1.3 Compose() [3/5]	450
16.44.1.4 Compose() [4/5]	450
16.44.1.5 Compose() [5/5]	450
16.44.2 Member Function Documentation	450
16.44.2.1 first_component()	450
16.44.2.2 second_component()	450
16.44.3 Friends And Related Function Documentation	450
16.44.3.1 operator<<	450
16.45 Simd128_impl< true, true, false, 2 >::Converter Union Reference	451
16.45.1 Field Documentation	451
16.45.1.1 v	451
16.45.1.2 t	451
16.46 Simd128_impl< true, true, false, 4 >::Converter Union Reference	451
16.46.1 Field Documentation	451
16.46.1.1 v	451
16.46.1.2 t	451
16.47 Simd128_impl< true, true, false, 8 >::Converter Union Reference	451
16.47.1 Field Documentation	451
16.47.1.1 v	452
16.47.1.2 t	452
16.48 Simd128_impl< true, true, true, 2 >::Converter Union Reference	452
16.48.1 Field Documentation	452
16.48.1.1 v	452

16.48.1.2 t . . . . .	452
16.49 Simd128_impl< true, true, true, 4 >::Converter Union Reference . . . . .	452
16.49.1 Field Documentation . . . . .	452
16.49.1.1 v . . . . .	452
16.49.1.2 t . . . . .	452
16.50 Simd128_impl< true, true, true, 8 >::Converter Union Reference . . . . .	453
16.50.1 Field Documentation . . . . .	453
16.50.1.1 v . . . . .	453
16.50.1.2 t . . . . .	453
16.51 Simd256_impl< true, false, true, 8 >::Converter Union Reference . . . . .	453
16.51.1 Field Documentation . . . . .	453
16.51.1.1 v . . . . .	453
16.51.1.2 t . . . . .	453
16.52 Simd256_impl< true, true, false, 2 >::Converter Union Reference . . . . .	453
16.52.1 Field Documentation . . . . .	453
16.52.1.1 v . . . . .	454
16.52.1.2 t . . . . .	454
16.53 Simd256_impl< true, true, false, 4 >::Converter Union Reference . . . . .	454
16.53.1 Field Documentation . . . . .	454
16.53.1.1 v . . . . .	454
16.53.1.2 t . . . . .	454
16.54 Simd256_impl< true, true, false, 8 >::Converter Union Reference . . . . .	454
16.54.1 Field Documentation . . . . .	454
16.54.1.1 v . . . . .	454
16.54.1.2 t . . . . .	454
16.55 Simd256_impl< true, true, true, 2 >::Converter Union Reference . . . . .	455
16.55.1 Field Documentation . . . . .	455
16.55.1.1 v . . . . .	455
16.55.1.2 t . . . . .	455
16.56 Simd256_impl< true, true, true, 4 >::Converter Union Reference . . . . .	455
16.56.1 Field Documentation . . . . .	455
16.56.1.1 v . . . . .	455
16.56.1.2 t . . . . .	455
16.57 Simd256_impl< true, true, true, 8 >::Converter Union Reference . . . . .	455
16.57.1 Field Documentation . . . . .	455
16.57.1.1 v . . . . .	456
16.57.1.2 t . . . . .	456
16.58 Simd512_impl< true, true, false, 8 >::Converter Union Reference . . . . .	456
16.58.1 Field Documentation . . . . .	456
16.58.1.1 v . . . . .	456
16.58.1.2 t . . . . .	456
16.59 Simd512_impl< true, true, true, 8 >::Converter Union Reference . . . . .	456

16.59.1 Field Documentation	456
16.59.1.1 v	456
16.59.1.2 t	456
16.60 ConvertTo< T > Struct Template Reference	457
16.60.1 Detailed Description	457
16.61 Coo< ValT, IdxT > Struct Template Reference	457
16.61.1 Member Typedef Documentation	457
16.61.1.1 Self	457
16.61.2 Constructor & Destructor Documentation	457
16.61.2.1 Coo() [1/4]	458
16.61.2.2 Coo() [2/4]	458
16.61.2.3 Coo() [3/4]	458
16.61.2.4 Coo() [4/4]	458
16.61.3 Member Function Documentation	458
16.61.3.1 operator=() [1/2]	458
16.61.3.2 operator=() [2/2]	458
16.61.4 Field Documentation	458
16.61.4.1 val	458
16.61.4.2 row	458
16.61.4.3 col	458
16.62 Coo< Field > Struct Template Reference	459
16.62.1 Constructor & Destructor Documentation	459
16.62.1.1 Coo() [1/4]	459
16.62.1.2 Coo() [2/4]	459
16.62.1.3 Coo() [3/4]	459
16.62.1.4 Coo() [4/4]	459
16.62.2 Member Function Documentation	459
16.62.2.1 operator=() [1/2]	459
16.62.2.2 operator=() [2/2]	460
16.62.3 Field Documentation	460
16.62.3.1 val	460
16.62.3.2 col	460
16.62.3.3 row	460
16.62.3.4 deleted	460
16.63 Coo< ValT, IdxT > Struct Template Reference	460
16.63.1 Member Typedef Documentation	460
16.63.1.1 Self	460
16.63.2 Constructor & Destructor Documentation	461
16.63.2.1 Coo() [1/4]	461
16.63.2.2 Coo() [2/4]	461
16.63.2.3 Coo() [3/4]	461
16.63.2.4 Coo() [4/4]	461

16.63.3 Member Function Documentation	461
16.63.3.1 operator=() [1/2]	461
16.63.3.2 operator=() [2/2]	461
16.63.4 Field Documentation	461
16.63.4.1 val	461
16.63.4.2 row	461
16.63.4.3 col	462
16.64 CooMat< Field > Struct Template Reference	462
16.64.1 Field Documentation	462
16.64.1.1 _coo16	462
16.64.1.2 _coo32	462
16.64.1.3 _coo64	462
16.64.1.4 _coo16_zo	462
16.64.1.5 _coo32_zo	462
16.64.1.6 _coo64_zo	462
16.65 count_nonconst_lvalue_reference< T > Struct Template Reference	463
16.66 count_nonconst_lvalue_reference< const T &, O... > Struct Template Reference	463
16.66.1 Field Documentation	463
16.66.1.1 n	463
16.67 count_nonconst_lvalue_reference< T &, O... > Struct Template Reference	463
16.67.1 Field Documentation	463
16.67.1.1 n	463
16.68 count_nonconst_lvalue_reference< T, O... > Struct Template Reference	463
16.68.1 Field Documentation	463
16.68.1.1 n	464
16.69 count_nonconst_lvalue_reference<> Struct Reference	464
16.69.1 Field Documentation	464
16.69.1.1 n	464
16.70 CsrMat< Field > Struct Template Reference	464
16.70.1 Field Documentation	464
16.70.1.1 _csr16	464
16.70.1.2 _csr32	464
16.70.1.3 _csr64	464
16.70.1.4 _csr16_zo	465
16.70.1.5 _csr32_zo	465
16.70.1.6 _csr64_zo	465
16.71 DefaultBoundedTag Struct Reference	465
16.71.1 Detailed Description	465
16.72 DefaultTag Struct Reference	465
16.72.1 Detailed Description	465
16.73 DelayedTag Struct Reference	465
16.73.1 Detailed Description	465

16.74 DivideAndConquer Struct Reference . . . . .	465
16.75 ElementTraits< Element > Struct Template Reference . . . . .	466
16.75.1 Detailed Description . . . . .	466
16.75.2 Member Typedef Documentation . . . . .	466
16.75.2.1 value . . . . .	466
16.76 ElementTraits< double > Struct Reference . . . . .	466
16.76.1 Member Typedef Documentation . . . . .	466
16.76.1.1 value . . . . .	466
16.77 ElementTraits< FFPACK::rns_double_elt > Struct Reference . . . . .	466
16.77.1 Member Typedef Documentation . . . . .	466
16.77.1.1 value . . . . .	467
16.78 ElementTraits< float > Struct Reference . . . . .	467
16.78.1 Member Typedef Documentation . . . . .	467
16.78.1.1 value . . . . .	467
16.79 ElementTraits< Givaro::Integer > Struct Reference . . . . .	467
16.79.1 Member Typedef Documentation . . . . .	467
16.79.1.1 value . . . . .	467
16.80 ElementTraits< int16_t > Struct Reference . . . . .	467
16.80.1 Member Typedef Documentation . . . . .	467
16.80.1.1 value . . . . .	468
16.81 ElementTraits< int32_t > Struct Reference . . . . .	468
16.81.1 Member Typedef Documentation . . . . .	468
16.81.1.1 value . . . . .	468
16.82 ElementTraits< int64_t > Struct Reference . . . . .	468
16.82.1 Member Typedef Documentation . . . . .	468
16.82.1.1 value . . . . .	468
16.83 ElementTraits< int8_t > Struct Reference . . . . .	468
16.83.1 Member Typedef Documentation . . . . .	468
16.83.1.1 value . . . . .	469
16.84 ElementTraits< Reclnt::rint< K > > Struct Template Reference . . . . .	469
16.84.1 Member Typedef Documentation . . . . .	469
16.84.1.1 value . . . . .	469
16.85 ElementTraits< Reclnt::rmint< K, MG > > Struct Template Reference . . . . .	469
16.85.1 Member Typedef Documentation . . . . .	469
16.85.1.1 value . . . . .	469
16.86 ElementTraits< Reclnt::ruint< K > > Struct Template Reference . . . . .	469
16.86.1 Member Typedef Documentation . . . . .	469
16.86.1.1 value . . . . .	470
16.87 ElementTraits< uint16_t > Struct Reference . . . . .	470
16.87.1 Member Typedef Documentation . . . . .	470
16.87.1.1 value . . . . .	470
16.88 ElementTraits< uint32_t > Struct Reference . . . . .	470

16.88.1 Member Typedef Documentation	470
16.88.1.1 value	470
16.89 ElementTraits< uint64_t > Struct Reference	470
16.89.1 Member Typedef Documentation	470
16.89.1.1 value	471
16.90 ElementTraits< uint8_t > Struct Reference	471
16.90.1 Member Typedef Documentation	471
16.90.1.1 value	471
16.91 EllMat< Field > Struct Template Reference	471
16.91.1 Field Documentation	471
16.91.1.1 _ell16	471
16.91.1.2 _ell32	471
16.91.1.3 _ell64	471
16.91.1.4 _ell16_zo	472
16.91.1.5 _ell32_zo	472
16.91.1.6 _ell64_zo	472
16.92 Failure Class Reference	472
16.92.1 Detailed Description	472
16.92.2 Constructor & Destructor Documentation	472
16.92.2.1 Failure()	472
16.92.3 Member Function Documentation	472
16.92.3.1 operator>() [1/2]	473
16.92.3.2 operator>() [2/2]	473
16.92.3.3 setErrorStream()	473
16.92.3.4 print()	473
16.92.4 Field Documentation	473
16.92.4.1 _errorStream	474
16.93 FailureCharpolyCheck Class Reference	474
16.94 FailureDetCheck Class Reference	474
16.95 FailureFgemmCheck Class Reference	474
16.96 FailureInvertCheck Class Reference	474
16.97 FailurePLUQCheck Class Reference	474
16.98 FailureTrsmCheck Class Reference	474
16.99 FieldSimd< _Field > Class Template Reference	474
16.99.1 Member Typedef Documentation	475
16.99.1.1 Field	475
16.99.1.2 Element	475
16.99.1.3 simd	476
16.99.1.4 vect_t	476
16.99.1.5 scalar_t	476
16.99.2 Constructor & Destructor Documentation	476
16.99.2.1 FieldSimd() [1/3]	476

16.99.2.2 FieldSimd() [2/3]	476
16.99.2.3 FieldSimd() [3/3]	476
16.99.3 Member Function Documentation	476
16.99.3.1 operator=() [1/2]	476
16.99.3.2 operator=() [2/2]	476
16.99.3.3 init() [1/2]	476
16.99.3.4 init() [2/2]	477
16.99.3.5 add() [1/2]	477
16.99.3.6 add() [2/2]	477
16.99.3.7 addin()	477
16.99.3.8 add_r() [1/2]	477
16.99.3.9 add_r() [2/2]	477
16.99.3.10 addin_r()	477
16.99.3.11 sub() [1/2]	477
16.99.3.12 sub() [2/2]	478
16.99.3.13 subin()	478
16.99.3.14 sub_r() [1/2]	478
16.99.3.15 sub_r() [2/2]	478
16.99.3.16 subin_r()	478
16.99.3.17 zero() [1/2]	478
16.99.3.18 zero() [2/2]	478
16.99.3.19 mod()	478
16.99.3.20 mul() [1/2]	478
16.99.3.21 mul() [2/2]	479
16.99.3.22 mulin()	479
16.99.3.23 mul_r() [1/2]	479
16.99.3.24 mul_r() [2/2]	479
16.99.3.25 axpy() [1/2]	479
16.99.3.26 axpy() [2/2]	479
16.99.3.27 axpyin()	479
16.99.3.28 axpy_r() [1/2]	479
16.99.3.29 axpy_r() [2/2]	480
16.99.3.30 axpyin_r()	480
16.99.3.31 maxpy() [1/2]	480
16.99.3.32 maxpy() [2/2]	480
16.99.3.33 maxpyin()	480
16.99.4 Field Documentation	480
16.99.4.1 vect_size	480
16.99.4.2 alignment	480
16.100 FieldTraits< Field > Struct Template Reference	481
16.100.1 Detailed Description	481
16.100.2 Member Typedef Documentation	481

16.100.2.1 category . . . . .	481
16.100.3 Field Documentation . . . . .	481
16.100.3.1 balanced . . . . .	481
16.101 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference . . . . .	481
16.101.1 Member Typedef Documentation . . . . .	481
16.101.1.1 category . . . . .	481
16.101.2 Field Documentation . . . . .	482
16.101.2.1 balanced . . . . .	482
16.102 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference . . . . .	482
16.102.1 Member Typedef Documentation . . . . .	482
16.102.1.1 category . . . . .	482
16.102.2 Field Documentation . . . . .	482
16.102.2.1 balanced . . . . .	482
16.103 FieldTraits< Givaro::Modular< Element > > Struct Template Reference . . . . .	482
16.103.1 Member Typedef Documentation . . . . .	483
16.103.1.1 category . . . . .	483
16.103.2 Field Documentation . . . . .	483
16.103.2.1 balanced . . . . .	483
16.104 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference . . . . .	483
16.104.1 Member Typedef Documentation . . . . .	483
16.104.1.1 category . . . . .	483
16.104.2 Field Documentation . . . . .	483
16.104.2.1 balanced . . . . .	483
16.105 FieldTraits< Givaro::ZRing< double > > Struct Reference . . . . .	483
16.105.1 Member Typedef Documentation . . . . .	484
16.105.1.1 category . . . . .	484
16.105.2 Field Documentation . . . . .	484
16.105.2.1 balanced . . . . .	484
16.106 FieldTraits< Givaro::ZRing< float > > Struct Reference . . . . .	484
16.106.1 Member Typedef Documentation . . . . .	484
16.106.1.1 category . . . . .	484
16.106.2 Field Documentation . . . . .	484
16.106.2.1 balanced . . . . .	484
16.107 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference . . . . .	484
16.107.1 Member Typedef Documentation . . . . .	485
16.107.1.1 category . . . . .	485
16.107.2 Field Documentation . . . . .	485
16.107.2.1 balanced . . . . .	485
16.108 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference . . . . .	485
16.108.1 Member Typedef Documentation . . . . .	485
16.108.1.1 category . . . . .	485
16.108.2 Field Documentation . . . . .	485

16.108.2.1 balanced	485
16.109 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference	486
16.109.1 Member Typedef Documentation	486
16.109.1.1 category	486
16.109.2 Field Documentation	486
16.109.2.1 balanced	486
16.110 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference	486
16.110.1 Member Typedef Documentation	486
16.110.1.1 category	486
16.110.2 Field Documentation	486
16.110.2.1 balanced	487
16.111 FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > Struct Template Reference	487
16.111.1 Member Typedef Documentation	487
16.111.1.1 category	487
16.111.2 Field Documentation	487
16.111.2.1 balanced	487
16.112 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference	487
16.112.1 Member Typedef Documentation	487
16.112.1.1 category	487
16.112.2 Field Documentation	488
16.112.2.1 balanced	488
16.113 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference	488
16.113.1 Member Typedef Documentation	488
16.113.1.1 category	488
16.113.2 Field Documentation	488
16.113.2.1 balanced	488
16.114 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference	488
16.114.1 Member Typedef Documentation	488
16.114.1.1 category	489
16.114.2 Field Documentation	489
16.114.2.1 balanced	489
16.115 Fixed Struct Reference	489
16.116 FixedPreclntTag Struct Reference	489
16.116.1 Detailed Description	489
16.117 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution Class Reference	489
16.117.1 Member Typedef Documentation	489
16.117.1.1 IntType	489
16.117.2 Constructor & Destructor Documentation	489
16.117.2.1 FloatingPointTestDistribution()	490
16.117.3 Member Function Documentation	490
16.117.3.1 operator>()()	490

16.118 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference . . . . .	490
16.118.1 Constructor & Destructor Documentation . . . . .	490
16.118.1.1 ForStrategy1D() [1/2] . . . . .	490
16.118.1.2 ForStrategy1D() [2/2] . . . . .	490
16.118.2 Member Function Documentation . . . . .	491
16.118.2.1 build() . . . . .	491
16.118.2.2 initialize() . . . . .	491
16.118.2.3 isTerminated() . . . . .	491
16.118.2.4 begin() . . . . .	491
16.118.2.5 end() . . . . .	491
16.118.2.6 numblocks() . . . . .	491
16.118.2.7 blockindex() . . . . .	491
16.118.2.8 operator++() . . . . .	491
16.118.3 Field Documentation . . . . .	491
16.118.3.1 ibeg . . . . .	491
16.118.3.2 iend . . . . .	491
16.118.3.3 current . . . . .	492
16.118.3.4 firstBlockSize . . . . .	492
16.118.3.5 lastBlockSize . . . . .	492
16.118.3.6 changeBS . . . . .	492
16.118.3.7 numBlock . . . . .	492
16.119 ForStrategy2D< blocksize_t, Cut, Param > Struct Template Reference . . . . .	492
16.119.1 Constructor & Destructor Documentation . . . . .	493
16.119.1.1 ForStrategy2D() . . . . .	493
16.119.2 Member Function Documentation . . . . .	493
16.119.2.1 initialize() . . . . .	493
16.119.2.2 isTerminated() . . . . .	493
16.119.2.3 ibegin() . . . . .	493
16.119.2.4 jbegin() . . . . .	493
16.119.2.5 iend() . . . . .	493
16.119.2.6 jend() . . . . .	493
16.119.2.7 operator++() . . . . .	493
16.119.2.8 rownumblocks() . . . . .	494
16.119.2.9 colnumblocks() . . . . .	494
16.119.2.10 blockindex() . . . . .	494
16.119.2.11 rowblockindex() . . . . .	494
16.119.2.12 colblockindex() . . . . .	494
16.119.3 Friends And Related Function Documentation . . . . .	494
16.119.3.1 operator<< . . . . .	494
16.119.4 Field Documentation . . . . .	494
16.119.4.1 _ibeg . . . . .	494
16.119.4.2 _iend . . . . .	494

16.119.4.3 <code>_jbeg</code>	494
16.119.4.4 <code>_jend</code>	494
16.119.4.5 <code>rowBlockSize</code>	495
16.119.4.6 <code>colBlockSize</code>	495
16.119.4.7 <code>current</code>	495
16.119.4.8 <code>lastRBS</code>	495
16.119.4.9 <code>lastCBS</code>	495
16.119.4.10 <code>changeRBS</code>	495
16.119.4.11 <code>changeCBS</code>	495
16.119.4.12 <code>numRowBlock</code>	495
16.119.4.13 <code>numColBlock</code>	495
16.119.4.14 <code>BLOCKS</code>	495
16.120 <code>ftmmLeftLowerNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	495
16.121 <code>ftmmLeftLowerNoTransUnit&lt; Element &gt;</code> Class Template Reference	496
16.122 <code>ftmmLeftLowerTransNonUnit&lt; Element &gt;</code> Class Template Reference	496
16.123 <code>ftmmLeftLowerTransUnit&lt; Element &gt;</code> Class Template Reference	496
16.124 <code>ftmmLeftUpperNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	496
16.125 <code>ftmmLeftUpperNoTransUnit&lt; Element &gt;</code> Class Template Reference	496
16.126 <code>ftmmLeftUpperTransNonUnit&lt; Element &gt;</code> Class Template Reference	496
16.127 <code>ftmmLeftUpperTransUnit&lt; Element &gt;</code> Class Template Reference	496
16.128 <code>ftmmRightLowerNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	496
16.129 <code>ftmmRightLowerNoTransUnit&lt; Element &gt;</code> Class Template Reference	497
16.130 <code>ftmmRightLowerTransNonUnit&lt; Element &gt;</code> Class Template Reference	497
16.131 <code>ftmmRightLowerTransUnit&lt; Element &gt;</code> Class Template Reference	497
16.132 <code>ftmmRightUpperNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	497
16.133 <code>ftmmRightUpperNoTransUnit&lt; Element &gt;</code> Class Template Reference	497
16.134 <code>ftmmRightUpperTransNonUnit&lt; Element &gt;</code> Class Template Reference	497
16.135 <code>ftmmRightUpperTransUnit&lt; Element &gt;</code> Class Template Reference	497
16.136 <code>frsmLeftLowerNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	497
16.137 <code>frsmLeftLowerNoTransUnit&lt; Element &gt;</code> Class Template Reference	498
16.138 <code>frsmLeftLowerTransNonUnit&lt; Element &gt;</code> Class Template Reference	498
16.139 <code>frsmLeftLowerTransUnit&lt; Element &gt;</code> Class Template Reference	498
16.140 <code>frsmLeftUpperNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	498
16.140.1 Detailed Description	498
16.141 <code>frsmLeftUpperNoTransUnit&lt; Element &gt;</code> Class Template Reference	498
16.142 <code>frsmLeftUpperTransNonUnit&lt; Element &gt;</code> Class Template Reference	499
16.143 <code>frsmLeftUpperTransUnit&lt; Element &gt;</code> Class Template Reference	499
16.144 <code>frsmRightLowerNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	499
16.145 <code>frsmRightLowerNoTransUnit&lt; Element &gt;</code> Class Template Reference	499
16.146 <code>frsmRightLowerTransNonUnit&lt; Element &gt;</code> Class Template Reference	499
16.147 <code>frsmRightLowerTransUnit&lt; Element &gt;</code> Class Template Reference	499
16.148 <code>frsmRightUpperNoTransNonUnit&lt; Element &gt;</code> Class Template Reference	499

16.149 frsmRightUpperNoTransUnit< Element > Class Template Reference . . . . .	499
16.150 frsmRightUpperTransNonUnit< Element > Class Template Reference . . . . .	500
16.151 frsmRightUpperTransUnit< Element > Class Template Reference . . . . .	500
16.152 GenericTag Struct Reference . . . . .	500
16.152.1 Detailed Description . . . . .	500
16.153 GenericTag Struct Reference . . . . .	500
16.153.1 Detailed Description . . . . .	500
16.154 Grain Struct Reference . . . . .	500
16.155 has_minus_eq_impl< C > Struct Template Reference . . . . .	500
16.155.1 Field Documentation . . . . .	500
16.155.1.1 value . . . . .	501
16.156 has_minus_impl< C > Struct Template Reference . . . . .	501
16.156.1 Field Documentation . . . . .	501
16.156.1.1 value . . . . .	501
16.157 has_mul_eq_impl< C > Struct Template Reference . . . . .	501
16.157.1 Field Documentation . . . . .	501
16.157.1.1 value . . . . .	501
16.158 has_mul_impl< C > Struct Template Reference . . . . .	501
16.158.1 Field Documentation . . . . .	501
16.158.1.1 value . . . . .	502
16.159 has_operation< T > Struct Template Reference . . . . .	502
16.159.1 Field Documentation . . . . .	502
16.159.1.1 value . . . . .	502
16.160 has_plus_eq_impl< C > Struct Template Reference . . . . .	502
16.160.1 Field Documentation . . . . .	502
16.160.1.1 value . . . . .	502
16.161 has_plus_impl< C > Struct Template Reference . . . . .	502
16.161.1 Field Documentation . . . . .	503
16.161.1.1 value . . . . .	503
16.162 HelperFlag Struct Reference . . . . .	503
16.162.1 Field Documentation . . . . .	503
16.162.1.1 none . . . . .	503
16.162.1.2 coo . . . . .	503
16.162.1.3 csr . . . . .	503
16.162.1.4 ell . . . . .	503
16.162.1.5 aut . . . . .	503
16.162.1.6 pm1 . . . . .	503
16.163 HelperMod< Field, ElementTraits > Struct Template Reference . . . . .	504
16.164 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference . . . . .	504
16.164.1 Constructor & Destructor Documentation . . . . .	504
16.164.1.1 HelperMod() [1/2] . . . . .	504
16.164.1.2 HelperMod() [2/2] . . . . .	504

16.164.2 Field Documentation	504
16.164.2.1 p	504
16.164.2.2 invp	504
16.164.2.3 min	504
16.164.2.4 max	504
16.164.2.5 pow50rem	505
16.165 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference	505
16.165.1 Constructor & Destructor Documentation	505
16.165.1.1 HelperMod() [1/2]	505
16.165.1.2 HelperMod() [2/2]	505
16.165.2 Field Documentation	505
16.165.2.1 p	505
16.166 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference	505
16.166.1 Constructor & Destructor Documentation	506
16.166.1.1 HelperMod() [1/2]	506
16.166.1.2 HelperMod() [2/2]	506
16.166.2 Field Documentation	506
16.166.2.1 p	506
16.167 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference	506
16.167.1 Constructor & Destructor Documentation	506
16.167.1.1 HelperMod() [1/2]	506
16.167.1.2 HelperMod() [2/2]	506
16.167.2 Field Documentation	506
16.167.2.1 p	507
16.167.2.2 invp	507
16.167.2.3 min	507
16.167.2.4 max	507
16.168 Hybrid Struct Reference	507
16.169 Info Struct Reference	507
16.169.1 Constructor & Destructor Documentation	507
16.169.1.1 Info() [1/4]	507
16.169.1.2 Info() [2/4]	508
16.169.1.3 Info() [3/4]	508
16.169.1.4 Info() [4/4]	508
16.169.2 Member Function Documentation	508
16.169.2.1 operator=() [1/2]	508
16.169.2.2 operator=() [2/2]	508
16.169.3 Field Documentation	508
16.169.3.1 size	508
16.169.3.2 perm	508
16.169.3.3 begin	508
16.170 Info Struct Reference	508

16.170.1 Constructor & Destructor Documentation	509
16.170.1.1 Info() [1/4]	509
16.170.1.2 Info() [2/4]	509
16.170.1.3 Info() [3/4]	509
16.170.1.4 Info() [4/4]	509
16.170.2 Member Function Documentation	509
16.170.2.1 operator=() [1/2]	509
16.170.2.2 operator=() [2/2]	509
16.170.3 Field Documentation	509
16.170.3.1 size	510
16.170.3.2 perm	510
16.170.3.3 begin	510
16.171 is_all_same< Args > Struct Template Reference	510
16.172 is_all_same< T, Args... > Struct Template Reference	510
16.172.1 Field Documentation	510
16.172.1.1 value	510
16.173 is_all_same<> Struct Reference	510
16.173.1 Field Documentation	510
16.173.1.1 value	510
16.174 is_simd< T > Struct Template Reference	511
16.174.1 Member Typedef Documentation	511
16.174.1.1 type	511
16.174.2 Field Documentation	511
16.174.2.1 value	511
16.175 isSparseMatrix< Field, M > Struct Template Reference	511
16.176 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference	511
16.177 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	512
16.178 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference	512
16.179 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference	512
16.180 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	513
16.181 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference	513
16.182 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference	513
16.183 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	513
16.184 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	514
16.185 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference	514
16.186 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference	514
16.187 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	515
16.188 isSparseMatrixMKLFormat< F, M > Struct Template Reference	515
16.189 isSparseMatrixSimdFormat< F, M > Struct Template Reference	515
16.190 isZOSparseMatrix< F, M > Struct Template Reference	515
16.191 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	516

16.192 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	516
16.193 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	516
16.194 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	517
16.195 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	517
16.196 Iterative Struct Reference	517
16.197 LazyTag Struct Reference	517
16.197.1 Detailed Description	517
16.198 limits< T > Struct Template Reference	518
16.199 limits< char > Struct Reference	518
16.199.1 Member Typedef Documentation	518
16.199.1.1 T	518
16.199.2 Member Function Documentation	518
16.199.2.1 max()	518
16.199.2.2 min()	518
16.199.2.3 digits()	518
16.200 limits< double > Struct Reference	518
16.200.1 Member Typedef Documentation	519
16.200.1.1 T	519
16.200.2 Member Function Documentation	519
16.200.2.1 max()	519
16.200.2.2 min()	519
16.200.2.3 digits()	519
16.201 limits< float > Struct Reference	519
16.201.1 Member Typedef Documentation	519
16.201.1.1 T	519
16.201.2 Member Function Documentation	520
16.201.2.1 max()	520
16.201.2.2 min()	520
16.201.2.3 digits()	520
16.202 limits< Givaro::Integer > Struct Reference	520
16.202.1 Member Typedef Documentation	520
16.202.1.1 T	520
16.202.2 Member Function Documentation	520
16.202.2.1 max()	520
16.202.2.2 min()	520
16.203 limits< int > Struct Reference	521
16.203.1 Member Typedef Documentation	521
16.203.1.1 T	521
16.203.2 Member Function Documentation	521
16.203.2.1 max()	521
16.203.2.2 min()	521

16.203.2.3 digits()	521
16.204 limits< long > Struct Reference	521
16.204.1 Member Typedef Documentation	522
16.204.1.1 T	522
16.204.2 Member Function Documentation	522
16.204.2.1 max()	522
16.204.2.2 min()	522
16.204.2.3 digits()	522
16.205 limits< long long > Struct Reference	522
16.205.1 Member Typedef Documentation	522
16.205.1.1 T	522
16.205.2 Member Function Documentation	522
16.205.2.1 max()	522
16.205.2.2 min()	523
16.205.2.3 digits()	523
16.206 limits< Reclnt::rint< K > > Struct Template Reference	523
16.206.1 Member Typedef Documentation	523
16.206.1.1 T	523
16.206.2 Member Function Documentation	523
16.206.2.1 max()	523
16.206.2.2 min()	523
16.207 limits< Reclnt::ruint< K > > Struct Template Reference	523
16.207.1 Member Typedef Documentation	524
16.207.1.1 T	524
16.207.2 Member Function Documentation	524
16.207.2.1 max()	524
16.207.2.2 min()	524
16.208 limits< short int > Struct Reference	524
16.208.1 Member Typedef Documentation	524
16.208.1.1 T	524
16.208.2 Member Function Documentation	524
16.208.2.1 max()	524
16.208.2.2 min()	525
16.208.2.3 digits()	525
16.209 limits< signed char > Struct Reference	525
16.209.1 Member Typedef Documentation	525
16.209.1.1 T	525
16.209.2 Member Function Documentation	525
16.209.2.1 max()	525
16.209.2.2 min()	525
16.209.2.3 digits()	525
16.210 limits< unsigned char > Struct Reference	525

16.210.1 Member Typedef Documentation . . . . .	526
16.210.1.1 T . . . . .	526
16.210.2 Member Function Documentation . . . . .	526
16.210.2.1 max() . . . . .	526
16.210.2.2 min() . . . . .	526
16.210.2.3 digits() . . . . .	526
16.211 limits< unsigned int > Struct Reference . . . . .	526
16.211.1 Member Typedef Documentation . . . . .	526
16.211.1.1 T . . . . .	527
16.211.2 Member Function Documentation . . . . .	527
16.211.2.1 max() . . . . .	527
16.211.2.2 min() . . . . .	527
16.211.2.3 digits() . . . . .	527
16.212 limits< unsigned long > Struct Reference . . . . .	527
16.212.1 Member Typedef Documentation . . . . .	527
16.212.1.1 T . . . . .	527
16.212.2 Member Function Documentation . . . . .	527
16.212.2.1 max() . . . . .	527
16.212.2.2 min() . . . . .	527
16.212.2.3 digits() . . . . .	528
16.213 limits< unsigned long long > Struct Reference . . . . .	528
16.213.1 Member Typedef Documentation . . . . .	528
16.213.1.1 T . . . . .	528
16.213.2 Member Function Documentation . . . . .	528
16.213.2.1 max() . . . . .	528
16.213.2.2 min() . . . . .	528
16.213.2.3 digits() . . . . .	528
16.214 limits< unsigned short int > Struct Reference . . . . .	528
16.214.1 Member Typedef Documentation . . . . .	529
16.214.1.1 T . . . . .	529
16.214.2 Member Function Documentation . . . . .	529
16.214.2.1 max() . . . . .	529
16.214.2.2 min() . . . . .	529
16.214.2.3 digits() . . . . .	529
16.215 MachineFloatTag Struct Reference . . . . .	529
16.215.1 Detailed Description . . . . .	529
16.216 MachineIntTag Struct Reference . . . . .	529
16.216.1 Detailed Description . . . . .	529
16.217 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference . . . . .	530
16.217.1 Member Typedef Documentation . . . . .	531
16.217.1.1 Self_t . . . . .	531
16.217.1.2 DelayedField_t . . . . .	531

16.217.1.3 DelayedField	531
16.217.1.4 DFElt	531
16.217.2 Constructor & Destructor Documentation	531
16.217.2.1 MMHelper() [1/5]	531
16.217.2.2 MMHelper() [2/5]	531
16.217.2.3 MMHelper() [3/5]	531
16.217.2.4 MMHelper() [4/5]	531
16.217.2.5 MMHelper() [5/5]	532
16.217.3 Member Function Documentation	532
16.217.3.1 initC()	532
16.217.3.2 initA()	532
16.217.3.3 initB()	532
16.217.3.4 initOut()	532
16.217.3.5 MaxDelayedDim()	532
16.217.3.6 Aunfit()	532
16.217.3.7 Bunfit()	532
16.217.3.8 setOutBounds()	532
16.217.3.9 checkA()	533
16.217.3.10 checkB()	533
16.217.3.11 checkOut() [1/2]	533
16.217.3.12 checkOut() [2/2]	533
16.217.4 Friends And Related Function Documentation	533
16.217.4.1 operator<<	533
16.217.5 Field Documentation	533
16.217.5.1 recLevel	533
16.217.5.2 FieldMin	534
16.217.5.3 FieldMax	534
16.217.5.4 Amin	534
16.217.5.5 Amax	534
16.217.5.6 Bmin	534
16.217.5.7 Bmax	534
16.217.5.8 Cmin	534
16.217.5.9 Cmax	534
16.217.5.10 Outmin	534
16.217.5.11 Outmax	534
16.217.5.12 MaxStorableValue	534
16.217.5.13 delayedField	534
16.217.5.14 parseq	535
16.218 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	535
16.218.1 Member Typedef Documentation	535
16.218.1.1 Self_t	535

16.218.2 Constructor & Destructor Documentation	535
16.218.2.1 MMHelper() [1/5]	535
16.218.2.2 MMHelper() [2/5]	536
16.218.2.3 MMHelper() [3/5]	536
16.218.2.4 MMHelper() [4/5]	536
16.218.2.5 MMHelper() [5/5]	536
16.218.3 Member Function Documentation	536
16.218.3.1 setNorm()	536
16.218.4 Friends And Related Function Documentation	536
16.218.4.1 operator<<	536
16.218.5 Field Documentation	536
16.218.5.1 normA	536
16.218.5.2 normB	537
16.218.5.3 recLevel	537
16.218.5.4 parseq	537
16.219 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeq← Trait > Struct Template Reference	537
16.219.1 Member Typedef Documentation	537
16.219.1.1 Self_t	537
16.219.2 Constructor & Destructor Documentation	538
16.219.2.1 MMHelper() [1/5]	538
16.219.2.2 MMHelper() [2/5]	538
16.219.2.3 MMHelper() [3/5]	538
16.219.2.4 MMHelper() [4/5]	538
16.219.2.5 MMHelper() [5/5]	538
16.219.3 Member Function Documentation	538
16.219.3.1 setNorm()	538
16.219.4 Friends And Related Function Documentation	538
16.219.4.1 operator<<	538
16.219.5 Field Documentation	539
16.219.5.1 normA	539
16.219.5.2 normB	539
16.219.5.3 recLevel	539
16.219.5.4 parseq	539
16.220 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Tem- plate Reference	539
16.220.1 Member Typedef Documentation	539
16.220.1.1 Self_t	539
16.220.2 Constructor & Destructor Documentation	540
16.220.2.1 MMHelper() [1/4]	540
16.220.2.2 MMHelper() [2/4]	540
16.220.2.3 MMHelper() [3/4]	540
16.220.2.4 MMHelper() [4/4]	540

16.220.3 Friends And Related Function Documentation	540
16.220.3.1 operator<<	540
16.220.4 Field Documentation	540
16.220.4.1 recLevel	540
16.220.4.2 parseq	540
16.221 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference	541
16.221.1 Member Typedef Documentation	541
16.221.1.1 Self_t	541
16.221.2 Constructor & Destructor Documentation	541
16.221.2.1 MMHelper() [1/5]	541
16.221.2.2 MMHelper() [2/5]	541
16.221.2.3 MMHelper() [3/5]	542
16.221.2.4 MMHelper() [4/5]	542
16.221.2.5 MMHelper() [5/5]	542
16.221.3 Member Function Documentation	542
16.221.3.1 setNorm()	542
16.221.4 Friends And Related Function Documentation	542
16.221.4.1 operator<<	542
16.221.5 Field Documentation	542
16.221.5.1 normA	542
16.221.5.2 normB	542
16.221.5.3 recLevel	543
16.221.5.4 parseq	543
16.222 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	543
16.222.1 Detailed Description	543
16.222.2 Member Typedef Documentation	543
16.222.2.1 Self_t	543
16.222.3 Constructor & Destructor Documentation	543
16.222.3.1 MMHelper() [1/4]	544
16.222.3.2 MMHelper() [2/4]	544
16.222.3.3 MMHelper() [3/4]	544
16.222.3.4 MMHelper() [4/4]	544
16.222.4 Friends And Related Function Documentation	544
16.222.4.1 operator<<	544
16.222.5 Field Documentation	544
16.222.5.1 recLevel	544
16.222.5.2 parseq	544
16.223 ModeTraits< Field > Struct Template Reference	544
16.223.1 Detailed Description	545
16.223.2 Member Typedef Documentation	545
16.223.2.1 value	545

16.224 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference . . . . .	545
16.224.1 Member Typedef Documentation . . . . .	545
16.224.1.1 value . . . . .	545
16.225 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference . . . . .	545
16.225.1 Member Typedef Documentation . . . . .	545
16.225.1.1 value . . . . .	546
16.226 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference . . . . .	546
16.226.1 Member Typedef Documentation . . . . .	546
16.226.1.1 value . . . . .	546
16.227 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference . . . . .	546
16.227.1 Member Typedef Documentation . . . . .	546
16.227.1.1 value . . . . .	546
16.228 ModeTraits< Givaro::Modular< int64_t, uint64_t > > Struct Reference . . . . .	546
16.228.1 Member Typedef Documentation . . . . .	547
16.228.1.1 value . . . . .	547
16.229 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference . . . . .	547
16.229.1 Member Typedef Documentation . . . . .	547
16.229.1.1 value . . . . .	547
16.230 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference . . . . .	547
16.230.1 Member Typedef Documentation . . . . .	547
16.230.1.1 value . . . . .	547
16.231 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference . . . . .	547
16.231.1 Member Typedef Documentation . . . . .	548
16.231.1.1 value . . . . .	548
16.232 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference . . . . .	548
16.232.1 Member Typedef Documentation . . . . .	548
16.232.1.1 value . . . . .	548
16.233 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference . . . . .	548
16.233.1 Member Typedef Documentation . . . . .	548
16.233.1.1 value . . . . .	548
16.234 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference . . . . .	549
16.234.1 Member Typedef Documentation . . . . .	549
16.234.1.1 value . . . . .	549
16.235 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference . . . . .	549
16.235.1 Member Typedef Documentation . . . . .	549
16.235.1.1 value . . . . .	549
16.236 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference . . . . .	549
16.236.1 Member Typedef Documentation . . . . .	549
16.236.1.1 value . . . . .	550
16.237 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference . . . . .	550
16.237.1 Member Typedef Documentation . . . . .	550
16.237.1.1 value . . . . .	550

16.238	ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference	550
16.238.1	Member Typedef Documentation	550
16.238.1.1	value	550
16.239	ModeTraits< Givaro::Montgomery< T > > Struct Template Reference	550
16.239.1	Member Typedef Documentation	551
16.239.1.1	value	551
16.240	ModeTraits< Givaro::ZRing< double > > Struct Reference	551
16.240.1	Member Typedef Documentation	551
16.240.1.1	value	551
16.241	ModeTraits< Givaro::ZRing< float > > Struct Reference	551
16.241.1	Member Typedef Documentation	551
16.241.1.1	value	551
16.242	ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference	551
16.242.1	Member Typedef Documentation	552
16.242.1.1	value	552
16.243	ModularBalanced< T > Class Template Reference	552
16.244	ModularTag Struct Reference	552
16.244.1	Detailed Description	552
16.245	Montgomery< T > Class Template Reference	552
16.246	need_field_characteristic< Field > Struct Template Reference	552
16.246.1	Field Documentation	552
16.246.1.1	value	552
16.247	need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference	552
16.247.1	Field Documentation	553
16.247.1.1	value	553
16.248	need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference	553
16.248.1	Field Documentation	553
16.248.1.1	value	553
16.249	NoSimd< T > Struct Template Reference	553
16.249.1	Member Typedef Documentation	553
16.249.1.1	vect_t	554
16.249.1.2	scalar_t	554
16.249.1.3	aligned_allocator	554
16.249.1.4	aligned_vector	554
16.249.1.5	is_same_element	554
16.249.2	Member Function Documentation	554
16.249.2.1	type_string()	554
16.249.2.2	valid()	554
16.249.2.3	compliant()	554
16.249.3	Field Documentation	554
16.249.3.1	vect_size	554
16.249.3.2	alignment	554

16.250 Parallel< C, P > Struct Template Reference . . . . .	555
16.250.1 Member Typedef Documentation . . . . .	555
16.250.1.1 Cut . . . . .	555
16.250.1.2 Param . . . . .	555
16.250.2 Constructor & Destructor Documentation . . . . .	555
16.250.2.1 Parallel() . . . . .	555
16.250.3 Member Function Documentation . . . . .	555
16.250.3.1 numthreads() . . . . .	555
16.250.3.2 set_numthreads() . . . . .	555
16.250.4 Friends And Related Function Documentation . . . . .	555
16.250.4.1 operator<< . . . . .	556
16.251 RNSInteger< RNS >::RandIter Class Reference . . . . .	556
16.251.1 Constructor & Destructor Documentation . . . . .	556
16.251.1.1 RandIter() . . . . .	556
16.251.2 Member Function Documentation . . . . .	556
16.251.2.1 random() [1/2] . . . . .	556
16.251.2.2 random() [2/2] . . . . .	557
16.251.2.3 operator>() [1/2] . . . . .	557
16.251.2.4 operator>() [2/2] . . . . .	557
16.251.2.5 ring() . . . . .	557
16.252 RNSIntegerMod< RNS >::RandIter Class Reference . . . . .	557
16.252.1 Constructor & Destructor Documentation . . . . .	557
16.252.1.1 RandIter() . . . . .	557
16.252.2 Member Function Documentation . . . . .	557
16.252.2.1 random() [1/2] . . . . .	558
16.252.2.2 random() [2/2] . . . . .	558
16.252.2.3 operator>() [1/2] . . . . .	558
16.252.2.4 operator>() [2/2] . . . . .	558
16.252.2.5 ring() . . . . .	558
16.253 readMyMachineType< Field, T > Struct Template Reference . . . . .	558
16.253.1 Member Typedef Documentation . . . . .	558
16.253.1.1 Element . . . . .	558
16.253.1.2 Element_ptr . . . . .	558
16.253.2 Member Function Documentation . . . . .	558
16.253.2.1 operator>() . . . . .	559
16.254 readMyMachineType< Field, mpz_t > Struct Template Reference . . . . .	559
16.254.1 Member Typedef Documentation . . . . .	559
16.254.1.1 Element . . . . .	559
16.254.1.2 Element_ptr . . . . .	559
16.254.2 Member Function Documentation . . . . .	559
16.254.2.1 operator>() . . . . .	559
16.255 Recursive Struct Reference . . . . .	560

16.256 Recursive Struct Reference . . . . .	560
16.257 rint< K > Class Template Reference . . . . .	560
16.258 rns_double Struct Reference . . . . .	560
16.258.1 Member Typedef Documentation . . . . .	561
16.258.1.1 integer . . . . .	561
16.258.1.2 ModField . . . . .	561
16.258.1.3 BasisElement . . . . .	561
16.258.1.4 Element . . . . .	561
16.258.1.5 Element_ptr . . . . .	561
16.258.1.6 ConstElement_ptr . . . . .	561
16.258.2 Constructor & Destructor Documentation . . . . .	561
16.258.2.1 rns_double() [1/4] . . . . .	562
16.258.2.2 rns_double() [2/4] . . . . .	562
16.258.2.3 rns_double() [3/4] . . . . .	562
16.258.2.4 rns_double() [4/4] . . . . .	562
16.258.3 Member Function Documentation . . . . .	562
16.258.3.1 precompute_cst() . . . . .	562
16.258.3.2 init() [1/3] . . . . .	562
16.258.3.3 init() [2/3] . . . . .	562
16.258.3.4 init_transpose() . . . . .	563
16.258.3.5 convert() [1/2] . . . . .	563
16.258.3.6 convert_transpose() . . . . .	563
16.258.3.7 reduce() . . . . .	563
16.258.3.8 init() [3/3] . . . . .	563
16.258.3.9 convert() [2/2] . . . . .	564
16.258.4 Field Documentation . . . . .	564
16.258.4.1 _basis . . . . .	564
16.258.4.2 _basisMax . . . . .	564
16.258.4.3 _negbasis . . . . .	564
16.258.4.4 _invbasis . . . . .	564
16.258.4.5 _field_rns . . . . .	564
16.258.4.6 _M . . . . .	564
16.258.4.7 _Mi . . . . .	565
16.258.4.8 _MMi . . . . .	565
16.258.4.9 _crt_in . . . . .	565
16.258.4.10 _crt_out . . . . .	565
16.258.4.11 _size . . . . .	565
16.258.4.12 _pbits . . . . .	565
16.258.4.13 _ldm . . . . .	565
16.258.4.14 _mi_sum . . . . .	565
16.259 rns_double_elt Struct Reference . . . . .	565
16.259.1 Constructor & Destructor Documentation . . . . .	566

16.259.1.1 rns_double_elt() [1/3]	566
16.259.1.2 ~rns_double_elt()	566
16.259.1.3 rns_double_elt() [2/3]	566
16.259.1.4 rns_double_elt() [3/3]	566
16.259.2 Member Function Documentation	566
16.259.2.1 operator&() [1/2]	566
16.259.2.2 operator&() [2/2]	566
16.259.3 Field Documentation	566
16.259.3.1 _ptr	567
16.259.3.2 _stride	567
16.259.3.3 _alloc	567
16.260 rns_double_elt_cstptr Struct Reference	567
16.260.1 Constructor & Destructor Documentation	568
16.260.1.1 rns_double_elt_cstptr() [1/5]	568
16.260.1.2 rns_double_elt_cstptr() [2/5]	568
16.260.1.3 rns_double_elt_cstptr() [3/5]	568
16.260.1.4 rns_double_elt_cstptr() [4/5]	568
16.260.1.5 rns_double_elt_cstptr() [5/5]	568
16.260.2 Member Function Documentation	568
16.260.2.1 operator&() [1/2]	568
16.260.2.2 operator*()	568
16.260.2.3 operator[]() [1/2]	568
16.260.2.4 operator[]() [2/2]	568
16.260.2.5 operator++()	569
16.260.2.6 operator--()	569
16.260.2.7 operator+()	569
16.260.2.8 operator-()	569
16.260.2.9 operator+=()	569
16.260.2.10 operator-=()	569
16.260.2.11 operator=()	569
16.260.2.12 operator<()	569
16.260.2.13 operator"!=(	569
16.260.2.14 operator&() [2/2]	569
16.260.3 Field Documentation	569
16.260.3.1 other	570
16.260.3.2 _ptr	570
16.260.3.3 _stride	570
16.260.3.4 _alloc	570
16.261 rns_double_elt_ptr Struct Reference	570
16.261.1 Constructor & Destructor Documentation	571
16.261.1.1 rns_double_elt_ptr() [1/5]	571
16.261.1.2 rns_double_elt_ptr() [2/5]	571

16.261.1.3 rns_double_elt_ptr() [3/5]	571
16.261.1.4 rns_double_elt_ptr() [4/5]	571
16.261.1.5 rns_double_elt_ptr() [5/5]	571
16.261.2 Member Function Documentation	571
16.261.2.1 operator&() [1/2]	571
16.261.2.2 operator*()	571
16.261.2.3 operator[]() [1/2]	571
16.261.2.4 operator[]() [2/2]	572
16.261.2.5 operator++()	572
16.261.2.6 operator--()	572
16.261.2.7 operator+()	572
16.261.2.8 operator-()	572
16.261.2.9 operator+=()	572
16.261.2.10 operator-=()	572
16.261.2.11 operator=()	572
16.261.2.12 operator<()	572
16.261.2.13 operator"!=(())	572
16.261.2.14 operator&() [2/2]	572
16.261.3 Field Documentation	573
16.261.3.1 other	573
16.261.3.2 _ptr	573
16.261.3.3 _stride	573
16.261.3.4 _alloc	573
16.262 rns_double_extended Struct Reference	573
16.262.1 Member Typedef Documentation	574
16.262.1.1 integer	574
16.262.1.2 ModField	574
16.262.1.3 BasisElement	574
16.262.1.4 Element	574
16.262.1.5 Element_ptr	574
16.262.1.6 ConstElement_ptr	574
16.262.2 Constructor & Destructor Documentation	574
16.262.2.1 rns_double_extended() [1/3]	575
16.262.2.2 rns_double_extended() [2/3]	575
16.262.2.3 rns_double_extended() [3/3]	575
16.262.3 Member Function Documentation	575
16.262.3.1 precompute_cst()	575
16.262.3.2 init() [1/3]	575
16.262.3.3 init() [2/3]	575
16.262.3.4 convert() [1/2]	576
16.262.3.5 init() [3/3]	576
16.262.3.6 convert() [2/2]	576

16.262.3.7 reduce()	576
16.262.4 Field Documentation	576
16.262.4.1 _basis	576
16.262.4.2 _basisMax	576
16.262.4.3 _negbasis	576
16.262.4.4 _invbasis	577
16.262.4.5 _field_rns	577
16.262.4.6 _M	577
16.262.4.7 _Mi	577
16.262.4.8 _MMi	577
16.262.4.9 _crt_in	577
16.262.4.10 _crt_out	577
16.262.4.11 _size	577
16.262.4.12 _pbits	577
16.262.4.13 _ldm	577
16.263 RNSElementTag Struct Reference	577
16.263.1 Detailed Description	578
16.264 RNSInteger< RNS > Class Template Reference	578
16.264.1 Member Typedef Documentation	579
16.264.1.1 BasisElement	579
16.264.1.2 integer	579
16.264.1.3 Element	579
16.264.1.4 Element_ptr	579
16.264.1.5 ConstElement_ptr	579
16.264.2 Constructor & Destructor Documentation	579
16.264.2.1 RNSInteger() [1/2]	579
16.264.2.2 RNSInteger() [2/2]	579
16.264.3 Member Function Documentation	579
16.264.3.1 rns()	579
16.264.3.2 size()	579
16.264.3.3 isOne()	580
16.264.3.4 isMOne()	580
16.264.3.5 isZero()	580
16.264.3.6 characteristic()	580
16.264.3.7 cardinality()	580
16.264.3.8 init() [1/2]	580
16.264.3.9 init() [2/2]	580
16.264.3.10 reduce() [1/2]	580
16.264.3.11 reduce() [2/2]	580
16.264.3.12 convert()	581
16.264.3.13 assign()	581
16.264.3.14 write() [1/2]	581

16.264.3.15 write() [2/2]	581
16.264.4 Field Documentation	581
16.264.4.1 _rns	581
16.264.4.2 one	581
16.264.4.3 mOne	581
16.264.4.4 zero	581
16.265 RNSIntegerMod< RNS > Class Template Reference	581
16.265.1 Member Typedef Documentation	583
16.265.1.1 Element	583
16.265.1.2 Element_ptr	583
16.265.1.3 ConstElement_ptr	583
16.265.1.4 BasisElement	583
16.265.1.5 ModField	583
16.265.1.6 integer	583
16.265.2 Constructor & Destructor Documentation	583
16.265.2.1 RNSIntegerMod()	583
16.265.3 Member Function Documentation	583
16.265.3.1 rns()	583
16.265.3.2 delayed()	584
16.265.3.3 size()	584
16.265.3.4 isOne()	584
16.265.3.5 isMOne()	584
16.265.3.6 isZero()	584
16.265.3.7 characteristic() [1/2]	584
16.265.3.8 characteristic() [2/2]	584
16.265.3.9 cardinality() [1/2]	584
16.265.3.10 cardinality() [2/2]	584
16.265.3.11 minElement()	584
16.265.3.12 maxElement()	584
16.265.3.13 init() [1/3]	585
16.265.3.14 init() [2/3]	585
16.265.3.15 reduce() [1/2]	585
16.265.3.16 reduce() [2/2]	585
16.265.3.17 init() [3/3]	585
16.265.3.18 convert()	585
16.265.3.19 assign()	585
16.265.3.20 add()	585
16.265.3.21 sub()	585
16.265.3.22 neg()	586
16.265.3.23 mul()	586
16.265.3.24 axpyin()	586
16.265.3.25 inv()	586

16.265.3.26 areEqual()	586
16.265.3.27 write() [1/2]	586
16.265.3.28 write() [2/2]	586
16.265.3.29 reduce_modp() [1/2]	586
16.265.3.30 write_matrix()	587
16.265.3.31 write_matrix_long()	587
16.265.3.32 reduce_modp() [2/2]	587
16.265.3.33 reduce_modp_rnsmajor()	587
16.265.4 Field Documentation	587
16.265.4.1 _p	587
16.265.4.2 _Mi_modp_rns	587
16.265.4.3 _iM_modp_rns	587
16.265.4.4 _rns	587
16.265.4.5 _F	588
16.265.4.6 _RNSdelayed	588
16.265.4.7 one	588
16.265.4.8 mOne	588
16.265.4.9 zero	588
16.266 rnsRandIter< RNS > Class Template Reference	588
16.266.1 Constructor & Destructor Documentation	588
16.266.1.1 rnsRandIter()	588
16.266.2 Member Function Documentation	589
16.266.2.1 random() [1/2]	589
16.266.2.2 operator>() [1/2]	589
16.266.2.3 operator>() [2/2]	589
16.266.2.4 random() [2/2]	589
16.266.2.5 ring()	589
16.267 Row Struct Reference	589
16.268 rint< K > Class Template Reference	589
16.269 ScalFunctions< Element > Struct Template Reference	589
16.269.1 Member Typedef Documentation	590
16.269.1.1 vectElt	590
16.269.2 Member Function Documentation	590
16.269.2.1 genInputs()	590
16.269.2.2 genInputsWithZero()	591
16.269.2.3 zero()	591
16.269.2.4 vand()	591
16.269.2.5 vor()	591
16.269.2.6 vxor()	591
16.269.2.7 vandnot()	591
16.269.2.8 add()	591
16.269.2.9 addin()	591

16.269.2.10 sub()	591
16.269.2.11 subin()	592
16.269.2.12 mul()	592
16.269.2.13 mulin()	592
16.269.2.14 div()	592
16.269.2.15 fmadd()	592
16.269.2.16 fmaddin()	592
16.269.2.17 fmsub()	592
16.269.2.18 fmsubin()	592
16.269.2.19 fnmadd()	593
16.269.2.20 fnmaddin()	593
16.269.2.21 lesser()	593
16.269.2.22 lesser_eq()	593
16.269.2.23 greater()	593
16.269.2.24 greater_eq()	593
16.269.2.25 eq()	593
16.269.2.26 unpacklo()	593
16.269.2.27 unpackhi()	594
16.269.2.28 unpacklohi()	594
16.269.2.29 pack_even()	594
16.269.2.30 pack_odd()	594
16.269.2.31 pack()	594
16.269.2.32 blend()	594
16.270 ScalFunctionsBase< Element, Enable > Struct Template Reference	594
16.271 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference	595
16.271.1 Member Function Documentation	595
16.271.1.1 get_default_random_generator()	595
16.271.1.2 ceil()	595
16.271.1.3 floor()	595
16.271.1.4 round()	595
16.271.1.5 blendv()	596
16.271.1.6 fma()	596
16.271.2 Field Documentation	596
16.271.2.1 _zero	596
16.271.2.2 cmp_true	596
16.271.2.3 cmp_false	596
16.272 ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference	596
16.272.1 Member Function Documentation	597
16.272.1.1 get_default_random_generator()	597
16.272.1.2 round()	597
16.272.1.3 fma()	597

16.272.1.4 mullo()	597
16.272.1.5 mulhi()	597
16.272.1.6 mulx()	597
16.272.1.7 fmaddx()	597
16.272.1.8 fmaddxin()	598
16.272.1.9 fmsubx()	598
16.272.1.10 fmsubxin()	598
16.272.1.11 fnmaddx()	598
16.272.1.12 fnmaddxin()	598
16.272.1.13 sra()	598
16.272.1.14 srl()	598
16.272.1.15 sll()	598
16.272.2 Field Documentation	598
16.272.2.1 _zero	599
16.272.2.2 cmp_true	599
16.272.2.3 cmp_false	599
16.273 Sequential Struct Reference	599
16.273.1 Constructor & Destructor Documentation	599
16.273.1.1 Sequential() [1/3]	599
16.273.1.2 Sequential() [2/3]	599
16.273.1.3 Sequential() [3/3]	599
16.273.2 Member Function Documentation	599
16.273.2.1 numthreads()	599
16.273.3 Friends And Related Function Documentation	600
16.273.3.1 operator<<	600
16.274 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference	600
16.275 Simd128_impl< true, false, true, 4 > Struct Reference	600
16.276 Simd128_impl< true, false, true, 8 > Struct Reference	600
16.277 Simd128_impl< true, true, false, 2 > Struct Reference	600
16.277.1 Member Typedef Documentation	602
16.277.1.1 scalar_t	602
16.277.1.2 aligned_allocator	602
16.277.1.3 aligned_vector	602
16.277.1.4 is_same_element	602
16.277.1.5 vect_t	602
16.277.2 Member Function Documentation	602
16.277.2.1 type_string()	603
16.277.2.2 set1()	603
16.277.2.3 set()	603
16.277.2.4 gather()	603
16.277.2.5 load()	603
16.277.2.6 loadu()	603

16.277.2.7 store()	603
16.277.2.8 storeu()	603
16.277.2.9 stream()	604
16.277.2.10 sra()	604
16.277.2.11 greater()	604
16.277.2.12 lesser()	604
16.277.2.13 greater_eq()	604
16.277.2.14 lesser_eq()	604
16.277.2.15 mulhi()	604
16.277.2.16 mulx()	604
16.277.2.17 fmaddx()	604
16.277.2.18 fmaddxin()	605
16.277.2.19 fnmaddx()	605
16.277.2.20 fnmaddxin()	605
16.277.2.21 fmsubx()	605
16.277.2.22 fmsubxin()	605
16.277.2.23 hadd_to_scal()	605
16.277.2.24 valid()	605
16.277.2.25 compliant()	606
16.277.2.26 sll()	606
16.277.2.27 srl()	606
16.277.2.28 shuffle()	606
16.277.2.29 unpacklo_intrinsic()	606
16.277.2.30 unpackhi_intrinsic()	606
16.277.2.31 unpacklo()	606
16.277.2.32 unpackhi()	606
16.277.2.33 unpacklohi()	606
16.277.2.34 pack_even()	607
16.277.2.35 pack_odd()	607
16.277.2.36 pack()	607
16.277.2.37 transpose()	607
16.277.2.38 blend()	607
16.277.2.39 add()	607
16.277.2.40 addin()	607
16.277.2.41 sub()	608
16.277.2.42 subin()	608
16.277.2.43 mullo()	608
16.277.2.44 mul()	608
16.277.2.45 fmadd()	608
16.277.2.46 fmaddin()	608
16.277.2.47 fnmadd()	608
16.277.2.48 fnmaddin()	608

16.277.2.49 fmsub()	609
16.277.2.50 fmsubin()	609
16.277.2.51 eq()	609
16.277.2.52 round()	609
16.277.2.53 mod()	609
16.277.2.54 zero()	609
16.277.2.55 sll128()	609
16.277.2.56 srl128()	609
16.277.2.57 vand()	610
16.277.2.58 vor()	610
16.277.2.59 vxor()	610
16.277.2.60 vandnot()	610
16.277.3 Field Documentation	610
16.277.3.1 vect_size	610
16.277.3.2 alignment	610
16.278 Simd128_impl< true, true, false, 4 > Struct Reference	610
16.278.1 Member Typedef Documentation	612
16.278.1.1 scalar_t	612
16.278.1.2 aligned_allocator	612
16.278.1.3 aligned_vector	612
16.278.1.4 is_same_element	612
16.278.1.5 vect_t	613
16.278.2 Member Function Documentation	613
16.278.2.1 type_string()	613
16.278.2.2 set1()	613
16.278.2.3 set()	613
16.278.2.4 gather()	613
16.278.2.5 load()	613
16.278.2.6 loadu()	613
16.278.2.7 store()	613
16.278.2.8 storeu()	613
16.278.2.9 stream()	614
16.278.2.10 sra()	614
16.278.2.11 greater()	614
16.278.2.12 lesser()	614
16.278.2.13 greater_eq()	614
16.278.2.14 lesser_eq()	614
16.278.2.15 mulhi()	614
16.278.2.16 mulx()	614
16.278.2.17 fmaddx()	614
16.278.2.18 fmaddxin()	615
16.278.2.19 fnmaddx()	615

16.278.2.20 fmaddxin()	615
16.278.2.21 fmsubx()	615
16.278.2.22 fmsubxin()	615
16.278.2.23 hadd_to_scal()	615
16.278.2.24 valid()	615
16.278.2.25 compliant()	616
16.278.2.26 sll()	616
16.278.2.27 srl()	616
16.278.2.28 shuffle()	616
16.278.2.29 unpacklo_intrinsic()	616
16.278.2.30 unpackhi_intrinsic()	616
16.278.2.31 unpacklo()	616
16.278.2.32 unpackhi()	616
16.278.2.33 unpacklohi()	616
16.278.2.34 pack_even()	617
16.278.2.35 pack_odd()	617
16.278.2.36 pack()	617
16.278.2.37 transpose()	617
16.278.2.38 blend()	617
16.278.2.39 add()	617
16.278.2.40 addin()	617
16.278.2.41 sub()	617
16.278.2.42 subin()	618
16.278.2.43 mullo()	618
16.278.2.44 mul()	618
16.278.2.45 fmadd()	618
16.278.2.46 fmaddin()	618
16.278.2.47 fnmadd()	618
16.278.2.48 fnmaddin()	618
16.278.2.49 fmsub()	618
16.278.2.50 fmsubin()	619
16.278.2.51 eq()	619
16.278.2.52 round()	619
16.278.2.53 mod()	619
16.278.2.54 zero()	619
16.278.2.55 sll128()	619
16.278.2.56 srl128()	619
16.278.2.57 vand()	619
16.278.2.58 vor()	620
16.278.2.59 vxor()	620
16.278.2.60 vandnot()	620
16.278.3 Field Documentation	620

16.278.3.1 vect_size . . . . .	620
16.278.3.2 alignment . . . . .	620
16.279 Simd128_impl< true, true, false, 8 > Struct Reference . . . . .	620
16.279.1 Member Typedef Documentation . . . . .	622
16.279.1.1 scalar_t . . . . .	622
16.279.1.2 aligned_allocator . . . . .	622
16.279.1.3 aligned_vector . . . . .	622
16.279.1.4 is_same_element . . . . .	623
16.279.1.5 vect_t . . . . .	623
16.279.2 Member Function Documentation . . . . .	623
16.279.2.1 type_string() . . . . .	623
16.279.2.2 set1() . . . . .	623
16.279.2.3 set() . . . . .	623
16.279.2.4 gather() . . . . .	623
16.279.2.5 load() . . . . .	623
16.279.2.6 loadu() . . . . .	623
16.279.2.7 store() . . . . .	623
16.279.2.8 storeu() . . . . .	624
16.279.2.9 stream() . . . . .	624
16.279.2.10 sra() . . . . .	624
16.279.2.11 greater() . . . . .	624
16.279.2.12 lesser() . . . . .	624
16.279.2.13 greater_eq() . . . . .	624
16.279.2.14 lesser_eq() . . . . .	624
16.279.2.15 mullo() . . . . .	624
16.279.2.16 mulhi() . . . . .	624
16.279.2.17 mulx() . . . . .	625
16.279.2.18 fmaddx() . . . . .	625
16.279.2.19 fmaddxin() . . . . .	625
16.279.2.20 fnmaddx() . . . . .	625
16.279.2.21 fnmaddxin() . . . . .	625
16.279.2.22 fmsubx() . . . . .	625
16.279.2.23 fmsubxin() . . . . .	625
16.279.2.24 hadd_to_scal() . . . . .	626
16.279.2.25 valid() . . . . .	626
16.279.2.26 compliant() . . . . .	626
16.279.2.27 get() . . . . .	626
16.279.2.28 sll() . . . . .	626
16.279.2.29 srl() . . . . .	626
16.279.2.30 shuffle() . . . . .	626
16.279.2.31 unpacklo_intrinsic() . . . . .	626
16.279.2.32 unpackhi_intrinsic() . . . . .	626

16.279.2.33 unpacklo()	627
16.279.2.34 unpackhi()	627
16.279.2.35 unpacklohi()	627
16.279.2.36 pack_even()	627
16.279.2.37 pack_odd()	627
16.279.2.38 pack()	627
16.279.2.39 transpose()	627
16.279.2.40 blend()	627
16.279.2.41 add()	628
16.279.2.42 addin()	628
16.279.2.43 sub()	628
16.279.2.44 subin()	628
16.279.2.45 mul()	628
16.279.2.46 fmadd()	628
16.279.2.47 fmaddin()	628
16.279.2.48 fnmadd()	628
16.279.2.49 fnmaddin()	629
16.279.2.50 fmsub()	629
16.279.2.51 fmsubin()	629
16.279.2.52 eq()	629
16.279.2.53 round()	629
16.279.2.54 mask_high()	629
16.279.2.55 mulhi_fast()	629
16.279.2.56 mod()	629
16.279.2.57 signbits()	630
16.279.2.58 zero()	630
16.279.2.59 sll128()	630
16.279.2.60 srl128()	630
16.279.2.61 vand()	630
16.279.2.62 vor()	630
16.279.2.63 vxor()	630
16.279.2.64 vandnot()	630
16.279.3 Field Documentation	630
16.279.3.1 vect_size	631
16.279.3.2 alignment	631
16.280 Simd128_impl< true, true, true, 2 > Struct Reference	631
16.280.1 Member Typedef Documentation	633
16.280.1.1 vect_t	633
16.280.1.2 scalar_t	633
16.280.1.3 aligned_allocator	633
16.280.1.4 aligned_vector	633
16.280.1.5 is_same_element	633

16.280.2 Member Function Documentation	633
16.280.2.1 type_string()	633
16.280.2.2 valid()	633
16.280.2.3 compliant()	633
16.280.2.4 set1()	633
16.280.2.5 set()	634
16.280.2.6 gather()	634
16.280.2.7 load()	634
16.280.2.8 loadu()	634
16.280.2.9 store()	634
16.280.2.10 storeu()	634
16.280.2.11 stream()	634
16.280.2.12 sll()	634
16.280.2.13 srl()	635
16.280.2.14 sra()	635
16.280.2.15 shuffle()	635
16.280.2.16 unpacklo_intrinsic()	635
16.280.2.17 unpackhi_intrinsic()	635
16.280.2.18 unpacklo()	635
16.280.2.19 unpackhi()	635
16.280.2.20 unpacklohi()	635
16.280.2.21 pack_even()	635
16.280.2.22 pack_odd()	636
16.280.2.23 pack()	636
16.280.2.24 transpose()	636
16.280.2.25 blend()	636
16.280.2.26 add()	636
16.280.2.27 addin()	636
16.280.2.28 sub()	636
16.280.2.29 subin()	637
16.280.2.30 mullo()	637
16.280.2.31 mul()	637
16.280.2.32 mulhi()	637
16.280.2.33 mulx()	637
16.280.2.34 fmadd()	637
16.280.2.35 fmaddin()	637
16.280.2.36 fmaddx()	637
16.280.2.37 fmaddxin()	638
16.280.2.38 fnmadd()	638
16.280.2.39 fnmaddin()	638
16.280.2.40 fnmaddx()	638
16.280.2.41 fnmaddxin()	638

16.280.2.42 fmsub()	638
16.280.2.43 fmsubin()	638
16.280.2.44 fmsubx()	638
16.280.2.45 fmsubxin()	639
16.280.2.46 eq()	639
16.280.2.47 greater()	639
16.280.2.48 lesser()	639
16.280.2.49 greater_eq()	639
16.280.2.50 lesser_eq()	639
16.280.2.51 hadd_to_scal()	639
16.280.2.52 round()	639
16.280.2.53 mod()	640
16.280.2.54 zero()	640
16.280.2.55 sll128()	640
16.280.2.56 srl128()	640
16.280.2.57 vand()	640
16.280.2.58 vor()	640
16.280.2.59 vxor()	640
16.280.2.60 vandnot()	640
16.280.3 Field Documentation	640
16.280.3.1 vect_size	641
16.280.3.2 alignment	641
16.281 Simd128_impl< true, true, true, 4 > Struct Reference	641
16.281.1 Member Typedef Documentation	643
16.281.1.1 vect_t	643
16.281.1.2 scalar_t	643
16.281.1.3 aligned_allocator	643
16.281.1.4 aligned_vector	643
16.281.1.5 is_same_element	643
16.281.2 Member Function Documentation	643
16.281.2.1 type_string()	643
16.281.2.2 valid()	643
16.281.2.3 compliant()	643
16.281.2.4 set1()	643
16.281.2.5 set()	644
16.281.2.6 gather()	644
16.281.2.7 load()	644
16.281.2.8 loadu()	644
16.281.2.9 store()	644
16.281.2.10 storeu()	644
16.281.2.11 stream()	644
16.281.2.12 sll()	644

16.281.2.13 srl()	644
16.281.2.14 sra()	645
16.281.2.15 shuffle()	645
16.281.2.16 unpacklo_intrinsic()	645
16.281.2.17 unpackhi_intrinsic()	645
16.281.2.18 unpacklo()	645
16.281.2.19 unpackhi()	645
16.281.2.20 unpacklohi()	645
16.281.2.21 pack_even()	645
16.281.2.22 pack_odd()	646
16.281.2.23 pack()	646
16.281.2.24 transpose()	646
16.281.2.25 blend()	646
16.281.2.26 add()	646
16.281.2.27 addin()	646
16.281.2.28 sub()	646
16.281.2.29 subin()	646
16.281.2.30 mullo()	647
16.281.2.31 mul()	647
16.281.2.32 mulhi()	647
16.281.2.33 mulx()	647
16.281.2.34 fmadd()	647
16.281.2.35 fmaddin()	647
16.281.2.36 fmaddx()	647
16.281.2.37 fmaddxin()	647
16.281.2.38 fnmadd()	648
16.281.2.39 fnmaddin()	648
16.281.2.40 fnmaddx()	648
16.281.2.41 fnmaddxin()	648
16.281.2.42 fmsub()	648
16.281.2.43 fmsubin()	648
16.281.2.44 fmsubx()	648
16.281.2.45 fmsubxin()	648
16.281.2.46 eq()	649
16.281.2.47 greater()	649
16.281.2.48 lesser()	649
16.281.2.49 greater_eq()	649
16.281.2.50 lesser_eq()	649
16.281.2.51 hadd_to_scal()	649
16.281.2.52 round()	649
16.281.2.53 mod()	649
16.281.2.54 zero()	650

16.281.2.55 sll128()	650
16.281.2.56 srl128()	650
16.281.2.57 vand()	650
16.281.2.58 vor()	650
16.281.2.59 vxor()	650
16.281.2.60 vandnot()	650
16.281.3 Field Documentation	650
16.281.3.1 vect_size	650
16.281.3.2 alignment	651
16.282 Simd128_impl< true, true, true, 8 > Struct Reference	651
16.282.1 Member Typedef Documentation	653
16.282.1.1 vect_t	653
16.282.1.2 scalar_t	653
16.282.1.3 aligned_allocator	653
16.282.1.4 aligned_vector	653
16.282.1.5 is_same_element	653
16.282.2 Member Function Documentation	653
16.282.2.1 type_string()	653
16.282.2.2 valid()	653
16.282.2.3 compliant()	653
16.282.2.4 set1()	654
16.282.2.5 set()	654
16.282.2.6 gather()	654
16.282.2.7 get()	654
16.282.2.8 load()	654
16.282.2.9 loadu()	654
16.282.2.10 store()	654
16.282.2.11 storeu()	654
16.282.2.12 stream()	654
16.282.2.13 sll()	655
16.282.2.14 srl()	655
16.282.2.15 sra()	655
16.282.2.16 shuffle()	655
16.282.2.17 unpacklo_intrinsic()	655
16.282.2.18 unpackhi_intrinsic()	655
16.282.2.19 unpacklo()	655
16.282.2.20 unpackhi()	655
16.282.2.21 unpacklohi()	655
16.282.2.22 pack_even()	656
16.282.2.23 pack_odd()	656
16.282.2.24 pack()	656
16.282.2.25 transpose()	656

16.282.2.26 blend()	656
16.282.2.27 add()	656
16.282.2.28 addin()	656
16.282.2.29 sub()	656
16.282.2.30 subin()	657
16.282.2.31 mullo()	657
16.282.2.32 mul()	657
16.282.2.33 mulhi()	657
16.282.2.34 mulx()	657
16.282.2.35 fmadd()	657
16.282.2.36 fmaddin()	657
16.282.2.37 fmaddx()	657
16.282.2.38 fmaddxin()	658
16.282.2.39 fnmadd()	658
16.282.2.40 fnmaddin()	658
16.282.2.41 fnmaddx()	658
16.282.2.42 fnmaddxin()	658
16.282.2.43 fmsub()	658
16.282.2.44 fmsubin()	658
16.282.2.45 fmsubx()	658
16.282.2.46 fmsubxin()	659
16.282.2.47 eq()	659
16.282.2.48 greater()	659
16.282.2.49 lesser()	659
16.282.2.50 greater_eq()	659
16.282.2.51 lesser_eq()	659
16.282.2.52 hadd_to_scal()	659
16.282.2.53 round()	659
16.282.2.54 mask_high()	660
16.282.2.55 mulhi_fast()	660
16.282.2.56 mod()	660
16.282.2.57 signbits()	660
16.282.2.58 zero()	660
16.282.2.59 sll128()	660
16.282.2.60 srl128()	660
16.282.2.61 vand()	660
16.282.2.62 vor()	661
16.282.2.63 vxor()	661
16.282.2.64 vandnot()	661
16.282.3 Field Documentation	661
16.282.3.1 vect_size	661
16.282.3.2 alignment	661

16.283 Simd128i_base Struct Reference . . . . .	661
16.283.1 Member Typedef Documentation . . . . .	662
16.283.1.1 vect_t . . . . .	662
16.283.2 Member Function Documentation . . . . .	662
16.283.2.1 zero() . . . . .	662
16.283.2.2 sll128() . . . . .	662
16.283.2.3 srl128() . . . . .	662
16.283.2.4 vand() . . . . .	662
16.283.2.5 vor() . . . . .	662
16.283.2.6 vxor() . . . . .	662
16.283.2.7 vandnot() . . . . .	663
16.284 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . . .	663
16.285 Simd256_impl< true, false, true, 4 > Struct Reference . . . . .	663
16.286 Simd256_impl< true, false, true, 8 > Struct Reference . . . . .	663
16.286.1 Member Typedef Documentation . . . . .	665
16.286.1.1 vect_t . . . . .	665
16.286.1.2 scalar_t . . . . .	665
16.286.1.3 aligned_allocator . . . . .	665
16.286.1.4 aligned_vector . . . . .	665
16.286.1.5 is_same_element . . . . .	665
16.286.2 Member Function Documentation . . . . .	665
16.286.2.1 type_string() . . . . .	665
16.286.2.2 valid() . . . . .	665
16.286.2.3 compliant() . . . . .	665
16.286.2.4 zero() . . . . .	665
16.286.2.5 set1() . . . . .	666
16.286.2.6 set() . . . . .	666
16.286.2.7 gather() . . . . .	666
16.286.2.8 load() . . . . .	666
16.286.2.9 loadu() . . . . .	666
16.286.2.10 store() . . . . .	666
16.286.2.11 storeu() . . . . .	666
16.286.2.12 stream() . . . . .	666
16.286.2.13 unpacklo_intrinsic() . . . . .	666
16.286.2.14 unpackhi_intrinsic() . . . . .	667
16.286.2.15 unpacklo() . . . . .	667
16.286.2.16 unpackhi() . . . . .	667
16.286.2.17 unpacklohi() . . . . .	667
16.286.2.18 pack_even() . . . . .	667
16.286.2.19 pack_odd() . . . . .	667
16.286.2.20 pack() . . . . .	667
16.286.2.21 transpose() . . . . .	667

16.286.2.22 blend()	668
16.286.2.23 blendv()	668
16.286.2.24 add()	668
16.286.2.25 addin()	668
16.286.2.26 sub()	668
16.286.2.27 subin()	668
16.286.2.28 mul()	668
16.286.2.29 mulin()	668
16.286.2.30 div()	669
16.286.2.31 fmadd()	669
16.286.2.32 fmaddin()	669
16.286.2.33 fnmadd()	669
16.286.2.34 fnmaddin()	669
16.286.2.35 fmsub()	669
16.286.2.36 fmsubin()	669
16.286.2.37 eq()	670
16.286.2.38 lesser()	670
16.286.2.39 lesser_eq()	670
16.286.2.40 greater()	670
16.286.2.41 greater_eq()	670
16.286.2.42 vand()	670
16.286.2.43 vor()	670
16.286.2.44 vxor()	670
16.286.2.45 vandnot()	671
16.286.2.46 floor()	671
16.286.2.47 ceil()	671
16.286.2.48 round()	671
16.286.2.49 hadd()	671
16.286.2.50 hadd_to_scal()	671
16.286.2.51 mod()	671
16.286.3 Field Documentation	671
16.286.3.1 vect_size	671
16.286.3.2 alignment	672
16.287 Simd256_impl< true, true, false, 2 > Struct Reference	672
16.287.1 Member Typedef Documentation	674
16.287.1.1 scalar_t	674
16.287.1.2 aligned_allocator	674
16.287.1.3 aligned_vector	674
16.287.1.4 is_same_element	674
16.287.1.5 simdHalf	674
16.287.1.6 vect_t	674
16.287.1.7 half_t	674

16.287.2 Member Function Documentation . . . . .	674
16.287.2.1 type_string() . . . . .	674
16.287.2.2 set1() . . . . .	674
16.287.2.3 set() . . . . .	674
16.287.2.4 gather() . . . . .	675
16.287.2.5 load() . . . . .	675
16.287.2.6 loadu() . . . . .	675
16.287.2.7 store() . . . . .	675
16.287.2.8 storeu() . . . . .	675
16.287.2.9 stream() . . . . .	675
16.287.2.10 sra() . . . . .	675
16.287.2.11 greater() . . . . .	676
16.287.2.12 lesser() . . . . .	676
16.287.2.13 greater_eq() . . . . .	676
16.287.2.14 lesser_eq() . . . . .	676
16.287.2.15 mulhi() . . . . .	676
16.287.2.16 mulx() . . . . .	676
16.287.2.17 fmaddx() . . . . .	676
16.287.2.18 fmaddxin() . . . . .	676
16.287.2.19 fnmaddx() . . . . .	677
16.287.2.20 fnmaddxin() . . . . .	677
16.287.2.21 fmsubx() . . . . .	677
16.287.2.22 fmsubxin() . . . . .	677
16.287.2.23 hadd_to_scal() . . . . .	677
16.287.2.24 valid() . . . . .	677
16.287.2.25 compliant() . . . . .	677
16.287.2.26 sll() . . . . .	677
16.287.2.27 srl() . . . . .	677
16.287.2.28 shuffle() . . . . .	678
16.287.2.29 unpacklo_intrinsic() . . . . .	678
16.287.2.30 unpackhi_intrinsic() . . . . .	678
16.287.2.31 unpacklo() . . . . .	678
16.287.2.32 unpackhi() . . . . .	678
16.287.2.33 unpacklohi() . . . . .	678
16.287.2.34 pack_even() . . . . .	678
16.287.2.35 pack_odd() . . . . .	678
16.287.2.36 pack() . . . . .	679
16.287.2.37 transpose() . . . . .	679
16.287.2.38 blend() [1/2] . . . . .	679
16.287.2.39 blend() [2/2] . . . . .	679
16.287.2.40 add() . . . . .	679
16.287.2.41 addin() . . . . .	679

16.287.2.42 sub()	680
16.287.2.43 subin()	680
16.287.2.44 mullo()	680
16.287.2.45 mul()	680
16.287.2.46 fmadd()	680
16.287.2.47 fmaddin()	680
16.287.2.48 fnmadd()	680
16.287.2.49 fnmaddin()	680
16.287.2.50 fmsub()	681
16.287.2.51 fmsubin()	681
16.287.2.52 eq()	681
16.287.2.53 round()	681
16.287.2.54 mod()	681
16.287.2.55 zero()	681
16.287.3 Field Documentation	681
16.287.3.1 vect_size	681
16.287.3.2 alignment	681
16.288 Simd256_impl< true, true, false, 4 > Struct Reference	682
16.288.1 Member Typedef Documentation	685
16.288.1.1 scalar_t [1/2]	685
16.288.1.2 aligned_allocator [1/2]	685
16.288.1.3 aligned_vector [1/2]	685
16.288.1.4 is_same_element [1/2]	685
16.288.1.5 simdHalf [1/2]	685
16.288.1.6 scalar_t [2/2]	685
16.288.1.7 aligned_allocator [2/2]	685
16.288.1.8 aligned_vector [2/2]	685
16.288.1.9 is_same_element [2/2]	686
16.288.1.10 simdHalf [2/2]	686
16.288.1.11 vect_t [1/2]	686
16.288.1.12 vect_t [2/2]	686
16.288.1.13 half_t [1/2]	686
16.288.1.14 half_t [2/2]	686
16.288.2 Member Function Documentation	686
16.288.2.1 type_string() [1/2]	686
16.288.2.2 set1() [1/2]	686
16.288.2.3 set() [1/3]	686
16.288.2.4 gather() [1/2]	687
16.288.2.5 load() [1/2]	687
16.288.2.6 loadu() [1/2]	687
16.288.2.7 store() [1/2]	687
16.288.2.8 storeu() [1/2]	687

16.288.2.9 stream() [1/2]	687
16.288.2.10 sra() [1/2]	687
16.288.2.11 greater() [1/2]	687
16.288.2.12 lesser() [1/2]	687
16.288.2.13 greater_eq() [1/2]	688
16.288.2.14 lesser_eq() [1/2]	688
16.288.2.15 mulhi() [1/2]	688
16.288.2.16 mulx() [1/2]	688
16.288.2.17 fmaddx() [1/2]	688
16.288.2.18 fmaddxin() [1/2]	688
16.288.2.19 fnmaddx() [1/2]	688
16.288.2.20 fnmaddxin() [1/2]	688
16.288.2.21 fmsubx() [1/2]	689
16.288.2.22 fmsubxin() [1/2]	689
16.288.2.23 hadd_to_scal() [1/2]	689
16.288.2.24 type_string() [2/2]	689
16.288.2.25 set1() [2/2]	689
16.288.2.26 set() [2/3]	689
16.288.2.27 gather() [2/2]	689
16.288.2.28 load() [2/2]	689
16.288.2.29 loadu() [2/2]	690
16.288.2.30 store() [2/2]	690
16.288.2.31 storeu() [2/2]	690
16.288.2.32 stream() [2/2]	690
16.288.2.33 sra() [2/2]	690
16.288.2.34 greater() [2/2]	690
16.288.2.35 lesser() [2/2]	690
16.288.2.36 greater_eq() [2/2]	690
16.288.2.37 lesser_eq() [2/2]	690
16.288.2.38 mulhi() [2/2]	691
16.288.2.39 mulx() [2/2]	691
16.288.2.40 fmaddx() [2/2]	691
16.288.2.41 fmaddxin() [2/2]	691
16.288.2.42 fnmaddx() [2/2]	691
16.288.2.43 fnmaddxin() [2/2]	691
16.288.2.44 fmsubx() [2/2]	691
16.288.2.45 fmsubxin() [2/2]	691
16.288.2.46 hadd_to_scal() [2/2]	692
16.288.2.47 valid() [1/2]	692
16.288.2.48 valid() [2/2]	692
16.288.2.49 compliant() [1/2]	692
16.288.2.50 compliant() [2/2]	692

16.288.2.51 set() [3/3]	692
16.288.2.52 sll() [1/2]	692
16.288.2.53 sll() [2/2]	693
16.288.2.54 srl() [1/2]	693
16.288.2.55 srl() [2/2]	693
16.288.2.56 shuffle_twice() [1/2]	693
16.288.2.57 shuffle_twice() [2/2]	693
16.288.2.58 shuffle() [1/2]	693
16.288.2.59 shuffle() [2/2]	693
16.288.2.60 unpacklo_intrinsic() [1/2]	693
16.288.2.61 unpacklo_intrinsic() [2/2]	693
16.288.2.62 unpackhi_intrinsic() [1/2]	694
16.288.2.63 unpackhi_intrinsic() [2/2]	694
16.288.2.64 unpacklo() [1/2]	694
16.288.2.65 unpacklo() [2/2]	694
16.288.2.66 unpackhi() [1/2]	694
16.288.2.67 unpackhi() [2/2]	694
16.288.2.68 unpacklohi() [1/2]	694
16.288.2.69 unpacklohi() [2/2]	694
16.288.2.70 pack_even() [1/2]	695
16.288.2.71 pack_even() [2/2]	695
16.288.2.72 pack_odd() [1/2]	695
16.288.2.73 pack_odd() [2/2]	695
16.288.2.74 pack() [1/2]	695
16.288.2.75 pack() [2/2]	695
16.288.2.76 transpose() [1/2]	695
16.288.2.77 transpose() [2/2]	696
16.288.2.78 blend() [1/2]	696
16.288.2.79 blend() [2/2]	696
16.288.2.80 add() [1/2]	696
16.288.2.81 add() [2/2]	696
16.288.2.82 addin() [1/2]	696
16.288.2.83 addin() [2/2]	696
16.288.2.84 sub() [1/2]	697
16.288.2.85 sub() [2/2]	697
16.288.2.86 subin() [1/2]	697
16.288.2.87 subin() [2/2]	697
16.288.2.88 mullo() [1/2]	697
16.288.2.89 mullo() [2/2]	697
16.288.2.90 mul() [1/2]	697
16.288.2.91 mul() [2/2]	697
16.288.2.92 fmadd() [1/2]	698

16.288.2.93 fmadd() [2/2]	698
16.288.2.94 fmaddin() [1/2]	698
16.288.2.95 fmaddin() [2/2]	698
16.288.2.96 fnmadd() [1/2]	698
16.288.2.97 fnmadd() [2/2]	698
16.288.2.98 fnmaddin() [1/2]	698
16.288.2.99 fnmaddin() [2/2]	698
16.288.2.100 fmsub() [1/2]	699
16.288.2.101 fmsub() [2/2]	699
16.288.2.102 fmsubin() [1/2]	699
16.288.2.103 fmsubin() [2/2]	699
16.288.2.104 eq() [1/2]	699
16.288.2.105 eq() [2/2]	699
16.288.2.106 round() [1/2]	699
16.288.2.107 round() [2/2]	700
16.288.2.108 mod() [1/2]	700
16.288.2.109 mod() [2/2]	700
16.288.2.110 zero() [1/2]	700
16.288.2.111 zero() [2/2]	700
16.288.2.112 vor()	700
16.288.2.113 vxor()	700
16.288.2.114 vand()	700
16.288.2.115 vandnot()	701
16.288.3 Field Documentation	701
16.288.3.1 vect_size	701
16.288.3.2 alignment	701
16.289 Simd256_impl< true, true, false, 8 > Struct Reference	701
16.289.1 Member Typedef Documentation	703
16.289.1.1 scalar_t	703
16.289.1.2 aligned_allocator	703
16.289.1.3 aligned_vector	703
16.289.1.4 is_same_element	703
16.289.1.5 simdHalf	703
16.289.1.6 vect_t	703
16.289.1.7 half_t	704
16.289.2 Member Function Documentation	704
16.289.2.1 type_string()	704
16.289.2.2 set1()	704
16.289.2.3 set()	704
16.289.2.4 gather()	704
16.289.2.5 load()	704
16.289.2.6 loadu()	704

16.289.2.7 store()	704
16.289.2.8 storeu()	704
16.289.2.9 stream()	705
16.289.2.10 sra()	705
16.289.2.11 greater()	705
16.289.2.12 lesser()	705
16.289.2.13 greater_eq()	705
16.289.2.14 lesser_eq()	705
16.289.2.15 mullo()	705
16.289.2.16 mulhi()	705
16.289.2.17 mulx()	705
16.289.2.18 fmaddx()	706
16.289.2.19 fmaddxin()	706
16.289.2.20 fnmaddx()	706
16.289.2.21 fnmaddxin()	706
16.289.2.22 fmsubx()	706
16.289.2.23 fmsubxin()	706
16.289.2.24 hadd_to_scal()	706
16.289.2.25 valid()	707
16.289.2.26 compliant()	707
16.289.2.27 get()	707
16.289.2.28 sll()	707
16.289.2.29 srl()	707
16.289.2.30 shuffle()	707
16.289.2.31 unpacklo_intrinsic()	707
16.289.2.32 unpackhi_intrinsic()	707
16.289.2.33 unpacklo()	707
16.289.2.34 unpackhi()	708
16.289.2.35 unpacklohi()	708
16.289.2.36 pack_even()	708
16.289.2.37 pack_odd()	708
16.289.2.38 pack()	708
16.289.2.39 transpose()	708
16.289.2.40 blend()	708
16.289.2.41 add()	708
16.289.2.42 addin()	709
16.289.2.43 sub()	709
16.289.2.44 subin()	709
16.289.2.45 mul()	709
16.289.2.46 fmadd()	709
16.289.2.47 fmaddin()	709
16.289.2.48 fnmadd()	709

16.289.2.49 fmaddin()	709
16.289.2.50 fmsub()	710
16.289.2.51 fmsubin()	710
16.289.2.52 eq()	710
16.289.2.53 round()	710
16.289.2.54 mask_high()	710
16.289.2.55 mulhi_fast()	710
16.289.2.56 mod()	710
16.289.2.57 signbits()	711
16.289.2.58 zero()	711
16.289.3 Field Documentation	711
16.289.3.1 vect_size	711
16.289.3.2 alignment	711
16.290 Simd256_impl< true, true, true, 2 > Struct Reference	711
16.290.1 Member Typedef Documentation	713
16.290.1.1 vect_t	713
16.290.1.2 half_t	713
16.290.1.3 scalar_t	713
16.290.1.4 simdHalf	713
16.290.1.5 aligned_allocator	713
16.290.1.6 aligned_vector	713
16.290.1.7 is_same_element	713
16.290.2 Member Function Documentation	714
16.290.2.1 type_string()	714
16.290.2.2 valid()	714
16.290.2.3 compliant()	714
16.290.2.4 set1()	714
16.290.2.5 set()	714
16.290.2.6 gather()	714
16.290.2.7 load()	714
16.290.2.8 loadu()	715
16.290.2.9 store()	715
16.290.2.10 storeu()	715
16.290.2.11 stream()	715
16.290.2.12 sll()	715
16.290.2.13 srl()	715
16.290.2.14 sra()	715
16.290.2.15 shuffle()	715
16.290.2.16 unpacklo_intrinsic()	715
16.290.2.17 unpackhi_intrinsic()	716
16.290.2.18 unpacklo()	716
16.290.2.19 unpackhi()	716

16.290.2.20 unpacklohi()	716
16.290.2.21 pack_even()	716
16.290.2.22 pack_odd()	716
16.290.2.23 pack()	716
16.290.2.24 transpose()	716
16.290.2.25 blend() [1/2]	717
16.290.2.26 blend() [2/2]	717
16.290.2.27 add()	717
16.290.2.28 addin()	717
16.290.2.29 sub()	717
16.290.2.30 subin()	717
16.290.2.31 mullo()	718
16.290.2.32 mul()	718
16.290.2.33 mulhi()	718
16.290.2.34 mulx()	718
16.290.2.35 fmadd()	718
16.290.2.36 fmaddin()	718
16.290.2.37 fmaddx()	718
16.290.2.38 fmaddxin()	718
16.290.2.39 fnmadd()	719
16.290.2.40 fnmaddin()	719
16.290.2.41 fnmaddx()	719
16.290.2.42 fnmaddxin()	719
16.290.2.43 fmsub()	719
16.290.2.44 fmsubin()	719
16.290.2.45 fmsubx()	719
16.290.2.46 fmsubxin()	719
16.290.2.47 eq()	720
16.290.2.48 greater()	720
16.290.2.49 lesser()	720
16.290.2.50 greater_eq()	720
16.290.2.51 lesser_eq()	720
16.290.2.52 hadd_to_scal()	720
16.290.2.53 round()	720
16.290.2.54 mod()	720
16.290.2.55 zero()	721
16.290.3 Field Documentation	721
16.290.3.1 vect_size	721
16.290.3.2 alignment	721
16.291 Simd256_impl< true, true, true, 4 > Struct Reference	721
16.291.1 Member Typedef Documentation	724
16.291.1.1 vect_t [1/2]	724

16.291.1.2 half_t [1/2]	724
16.291.1.3 scalar_t [1/2]	725
16.291.1.4 simdHalf [1/2]	725
16.291.1.5 aligned_allocator [1/2]	725
16.291.1.6 aligned_vector [1/2]	725
16.291.1.7 is_same_element [1/2]	725
16.291.1.8 vect_t [2/2]	725
16.291.1.9 half_t [2/2]	725
16.291.1.10 scalar_t [2/2]	725
16.291.1.11 simdHalf [2/2]	725
16.291.1.12 aligned_allocator [2/2]	725
16.291.1.13 aligned_vector [2/2]	725
16.291.1.14 is_same_element [2/2]	725
16.291.2 Member Function Documentation	726
16.291.2.1 type_string() [1/2]	726
16.291.2.2 valid() [1/2]	726
16.291.2.3 compliant() [1/2]	726
16.291.2.4 set1() [1/2]	726
16.291.2.5 set() [1/2]	726
16.291.2.6 gather() [1/2]	726
16.291.2.7 load() [1/2]	726
16.291.2.8 loadu() [1/2]	726
16.291.2.9 store() [1/2]	727
16.291.2.10 storeu() [1/2]	727
16.291.2.11 stream() [1/2]	727
16.291.2.12 sll() [1/2]	727
16.291.2.13 srl() [1/2]	727
16.291.2.14 sra() [1/2]	727
16.291.2.15 shuffle_twice() [1/2]	727
16.291.2.16 shuffle() [1/2]	727
16.291.2.17 unpacklo_intrinsic() [1/2]	727
16.291.2.18 unpackhi_intrinsic() [1/2]	728
16.291.2.19 unpacklo() [1/2]	728
16.291.2.20 unpackhi() [1/2]	728
16.291.2.21 unpacklohi() [1/2]	728
16.291.2.22 pack_even() [1/2]	728
16.291.2.23 pack_odd() [1/2]	728
16.291.2.24 pack() [1/2]	728
16.291.2.25 transpose() [1/2]	728
16.291.2.26 blend() [1/2]	729
16.291.2.27 add() [1/2]	729
16.291.2.28 addin() [1/2]	729

16.291.2.29 sub() [1/2]	729
16.291.2.30 subin() [1/2]	729
16.291.2.31 mullo() [1/2]	729
16.291.2.32 mul() [1/2]	729
16.291.2.33 mulhi() [1/2]	729
16.291.2.34 mulx() [1/2]	730
16.291.2.35 fmadd() [1/2]	730
16.291.2.36 fmaddin() [1/2]	730
16.291.2.37 fmaddx() [1/2]	730
16.291.2.38 fmaddxin() [1/2]	730
16.291.2.39 fnmadd() [1/2]	730
16.291.2.40 fnmaddin() [1/2]	730
16.291.2.41 fnmaddx() [1/2]	731
16.291.2.42 fnmaddxin() [1/2]	731
16.291.2.43 fmsub() [1/2]	731
16.291.2.44 fmsubin() [1/2]	731
16.291.2.45 fmsubx() [1/2]	731
16.291.2.46 fmsubxin() [1/2]	731
16.291.2.47 eq() [1/2]	731
16.291.2.48 greater() [1/2]	731
16.291.2.49 lesser() [1/2]	732
16.291.2.50 greater_eq() [1/2]	732
16.291.2.51 lesser_eq() [1/2]	732
16.291.2.52 hadd_to_scal() [1/2]	732
16.291.2.53 round() [1/2]	732
16.291.2.54 mod() [1/2]	732
16.291.2.55 type_string() [2/2]	732
16.291.2.56 valid() [2/2]	732
16.291.2.57 compliant() [2/2]	733
16.291.2.58 set1() [2/2]	733
16.291.2.59 set() [2/2]	733
16.291.2.60 gather() [2/2]	733
16.291.2.61 load() [2/2]	733
16.291.2.62 loadu() [2/2]	733
16.291.2.63 store() [2/2]	733
16.291.2.64 storeu() [2/2]	734
16.291.2.65 stream() [2/2]	734
16.291.2.66 sll() [2/2]	734
16.291.2.67 srl() [2/2]	734
16.291.2.68 sra() [2/2]	734
16.291.2.69 shuffle_twice() [2/2]	734
16.291.2.70 shuffle() [2/2]	734

16.291.2.71 unpacklo_intrinsic() [2/2]	734
16.291.2.72 unpackhi_intrinsic() [2/2]	734
16.291.2.73 unpacklo() [2/2]	735
16.291.2.74 unpackhi() [2/2]	735
16.291.2.75 unpacklohi() [2/2]	735
16.291.2.76 pack_even() [2/2]	735
16.291.2.77 pack_odd() [2/2]	735
16.291.2.78 pack() [2/2]	735
16.291.2.79 transpose() [2/2]	735
16.291.2.80 blend() [2/2]	736
16.291.2.81 add() [2/2]	736
16.291.2.82 addin() [2/2]	736
16.291.2.83 sub() [2/2]	736
16.291.2.84 subin() [2/2]	736
16.291.2.85 mullo() [2/2]	736
16.291.2.86 mul() [2/2]	736
16.291.2.87 mulhi() [2/2]	737
16.291.2.88 mulx() [2/2]	737
16.291.2.89 fmadd() [2/2]	737
16.291.2.90 fmaddin() [2/2]	737
16.291.2.91 fmaddx() [2/2]	737
16.291.2.92 fmaddxin() [2/2]	737
16.291.2.93 fnmadd() [2/2]	737
16.291.2.94 fnmaddin() [2/2]	737
16.291.2.95 fnmaddx() [2/2]	738
16.291.2.96 fnmaddxin() [2/2]	738
16.291.2.97 fmsub() [2/2]	738
16.291.2.98 fmsubin() [2/2]	738
16.291.2.99 fmsubx() [2/2]	738
16.291.2.100 fmsubxin() [2/2]	738
16.291.2.101 eq() [2/2]	738
16.291.2.102 greater() [2/2]	739
16.291.2.103 lesser() [2/2]	739
16.291.2.104 greater_eq() [2/2]	739
16.291.2.105 lesser_eq() [2/2]	739
16.291.2.106 hadd_to_scal() [2/2]	739
16.291.2.107 round() [2/2]	739
16.291.2.108 mod() [2/2]	739
16.291.2.109 zero() [1/2]	739
16.291.2.110 zero() [2/2]	740
16.291.2.111 vor()	740
16.291.2.112 vxor()	740

16.291.2.113 vand()	740
16.291.2.114 vandnot()	740
16.291.3 Field Documentation	740
16.291.3.1 vect_size	740
16.291.3.2 alignment	740
16.292 Simd256_impl< true, true, true, 8 > Struct Reference	740
16.292.1 Member Typedef Documentation	742
16.292.1.1 vect_t	742
16.292.1.2 half_t	743
16.292.1.3 scalar_t	743
16.292.1.4 simdHalf	743
16.292.1.5 aligned_allocator	743
16.292.1.6 aligned_vector	743
16.292.1.7 is_same_element	743
16.292.2 Member Function Documentation	743
16.292.2.1 type_string()	743
16.292.2.2 valid()	743
16.292.2.3 compliant()	743
16.292.2.4 set1()	743
16.292.2.5 set()	744
16.292.2.6 gather()	744
16.292.2.7 get()	744
16.292.2.8 load()	744
16.292.2.9 loadu()	744
16.292.2.10 store()	744
16.292.2.11 storeu()	744
16.292.2.12 stream()	744
16.292.2.13 sll()	744
16.292.2.14 srl()	745
16.292.2.15 sra()	745
16.292.2.16 shuffle()	745
16.292.2.17 unpacklo_intrinsic()	745
16.292.2.18 unpackhi_intrinsic()	745
16.292.2.19 unpacklo()	745
16.292.2.20 unpackhi()	745
16.292.2.21 unpacklohi()	745
16.292.2.22 pack_even()	745
16.292.2.23 pack_odd()	746
16.292.2.24 pack()	746
16.292.2.25 transpose()	746
16.292.2.26 blend()	746
16.292.2.27 add()	746

16.292.2.28 addin()	746
16.292.2.29 sub()	746
16.292.2.30 subin()	746
16.292.2.31 mullo()	747
16.292.2.32 mul()	747
16.292.2.33 mulhi()	747
16.292.2.34 mulx()	747
16.292.2.35 fmadd()	747
16.292.2.36 fmaddin()	747
16.292.2.37 fmaddx()	747
16.292.2.38 fmaddxin()	747
16.292.2.39 fnmadd()	748
16.292.2.40 fnmaddin()	748
16.292.2.41 fnmaddx()	748
16.292.2.42 fnmaddxin()	748
16.292.2.43 fmsub()	748
16.292.2.44 fmsubin()	748
16.292.2.45 fmsubx()	748
16.292.2.46 fmsubxin()	749
16.292.2.47 eq()	749
16.292.2.48 greater()	749
16.292.2.49 lesser()	749
16.292.2.50 greater_eq()	749
16.292.2.51 lesser_eq()	749
16.292.2.52 hadd_to_scal()	749
16.292.2.53 round()	749
16.292.2.54 mask_high()	749
16.292.2.55 mulhi_fast()	750
16.292.2.56 mod()	750
16.292.2.57 signbits()	750
16.292.2.58 zero()	750
16.292.3 Field Documentation	750
16.292.3.1 vect_size	750
16.292.3.2 alignment	750
16.293 Simd256fp_base Struct Reference	750
16.294 Simd256i_base Struct Reference	751
16.294.1 Member Typedef Documentation	751
16.294.1.1 vect_t	751
16.294.2 Member Function Documentation	751
16.294.2.1 zero()	751
16.295 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference	751
16.296 Simd512_impl< true, false, true, 4 > Struct Reference	751

16.297 Simd512_impl< true, false, true, 8 > Struct Reference . . . . .	751
16.297.1 Member Typedef Documentation . . . . .	753
16.297.1.1 vect_t . . . . .	753
16.297.1.2 scalar_t . . . . .	753
16.297.1.3 aligned_allocator . . . . .	753
16.297.1.4 aligned_vector . . . . .	753
16.297.1.5 is_same_element . . . . .	753
16.297.2 Member Function Documentation . . . . .	753
16.297.2.1 type_string() . . . . .	753
16.297.2.2 valid() . . . . .	753
16.297.2.3 compliant() . . . . .	753
16.297.2.4 zero() . . . . .	754
16.297.2.5 set1() . . . . .	754
16.297.2.6 set() . . . . .	754
16.297.2.7 gather() . . . . .	754
16.297.2.8 load() . . . . .	754
16.297.2.9 loadu() . . . . .	754
16.297.2.10 store() . . . . .	754
16.297.2.11 storeu() . . . . .	754
16.297.2.12 stream() . . . . .	755
16.297.2.13 shuffle() . . . . .	755
16.297.2.14 unpacklo_intrinsic() . . . . .	755
16.297.2.15 unpackhi_intrinsic() . . . . .	755
16.297.2.16 unpacklo() . . . . .	755
16.297.2.17 unpackhi() . . . . .	755
16.297.2.18 unpacklohi() . . . . .	755
16.297.2.19 pack_even() . . . . .	755
16.297.2.20 pack_odd() . . . . .	756
16.297.2.21 pack() . . . . .	756
16.297.2.22 transpose() . . . . .	756
16.297.2.23 blend() . . . . .	756
16.297.2.24 blendv() . . . . .	756
16.297.2.25 add() . . . . .	756
16.297.2.26 addin() . . . . .	756
16.297.2.27 sub() . . . . .	757
16.297.2.28 subin() . . . . .	757
16.297.2.29 mul() . . . . .	757
16.297.2.30 mulin() . . . . .	757
16.297.2.31 div() . . . . .	757
16.297.2.32 fmadd() . . . . .	757
16.297.2.33 fmaddin() . . . . .	757
16.297.2.34 fnmadd() . . . . .	757

16.297.2.35 fnmaddin()	758
16.297.2.36 fmsub()	758
16.297.2.37 fmsubin()	758
16.297.2.38 eq()	758
16.297.2.39 lesser()	758
16.297.2.40 lesser_eq()	758
16.297.2.41 greater()	758
16.297.2.42 greater_eq()	758
16.297.2.43 floor()	759
16.297.2.44 ceil()	759
16.297.2.45 round()	759
16.297.2.46 hadd()	759
16.297.2.47 hadd_to_scal()	759
16.297.3 Field Documentation	759
16.297.3.1 vect_size	759
16.297.3.2 alignment	759
16.298 Simd512_impl< true, true, false, 8 > Struct Reference	759
16.298.1 Member Typedef Documentation	761
16.298.1.1 scalar_t	761
16.298.1.2 aligned_allocator	762
16.298.1.3 aligned_vector	762
16.298.1.4 is_same_element	762
16.298.1.5 simdHalf	762
16.298.1.6 vect_t	762
16.298.1.7 half_t	762
16.298.2 Member Function Documentation	762
16.298.2.1 type_string()	762
16.298.2.2 set1()	762
16.298.2.3 set() [1/2]	762
16.298.2.4 gather()	763
16.298.2.5 load()	763
16.298.2.6 loadu()	763
16.298.2.7 store()	763
16.298.2.8 maskstore()	763
16.298.2.9 storeu()	763
16.298.2.10 stream()	763
16.298.2.11 sra()	763
16.298.2.12 greater()	763
16.298.2.13 lesser()	764
16.298.2.14 greater_eq()	764
16.298.2.15 lesser_eq()	764
16.298.2.16 mullo()	764

16.298.2.17 mulhi()	764
16.298.2.18 mulx()	764
16.298.2.19 fmaddx()	764
16.298.2.20 fmaddxin()	764
16.298.2.21 fnmaddx()	765
16.298.2.22 fnmaddxin()	765
16.298.2.23 fmsubx()	765
16.298.2.24 fmsubxin()	765
16.298.2.25 hadd_to_scal()	765
16.298.2.26 valid()	765
16.298.2.27 compliant()	765
16.298.2.28 set() [2/2]	765
16.298.2.29 sll()	766
16.298.2.30 srl()	766
16.298.2.31 shuffle()	766
16.298.2.32 unpacklo_intrinsic()	766
16.298.2.33 unpackhi_intrinsic()	766
16.298.2.34 unpacklo()	766
16.298.2.35 unpackhi()	766
16.298.2.36 unpacklohi()	766
16.298.2.37 pack_even()	766
16.298.2.38 pack_odd()	767
16.298.2.39 pack()	767
16.298.2.40 transpose()	767
16.298.2.41 blend()	767
16.298.2.42 add()	767
16.298.2.43 addin()	767
16.298.2.44 sub()	767
16.298.2.45 subin()	768
16.298.2.46 mul()	768
16.298.2.47 fmadd()	768
16.298.2.48 fmaddin()	768
16.298.2.49 fnmadd()	768
16.298.2.50 fnmaddin()	768
16.298.2.51 fmsub()	768
16.298.2.52 fmsubin()	768
16.298.2.53 eq()	769
16.298.2.54 round()	769
16.298.2.55 mask_high()	769
16.298.2.56 mulhi_fast()	769
16.298.2.57 mod()	769
16.298.2.58 signbits()	769

16.298.2.59 zero()	769
16.298.2.60 vor()	769
16.298.2.61 vxor()	770
16.298.2.62 vand()	770
16.298.2.63 vandnot()	770
16.298.3 Field Documentation	770
16.298.3.1 vect_size	770
16.298.3.2 alignment	770
16.299 Simd512_impl< true, true, true, 8 > Struct Reference	770
16.299.1 Member Typedef Documentation	772
16.299.1.1 vect_t	772
16.299.1.2 half_t	772
16.299.1.3 scalar_t	772
16.299.1.4 simdHalf	772
16.299.1.5 aligned_allocator	773
16.299.1.6 aligned_vector	773
16.299.1.7 is_same_element	773
16.299.2 Member Function Documentation	773
16.299.2.1 type_string()	773
16.299.2.2 valid()	773
16.299.2.3 compliant()	773
16.299.2.4 set1()	773
16.299.2.5 set() [1/2]	773
16.299.2.6 set() [2/2]	773
16.299.2.7 gather()	774
16.299.2.8 load()	774
16.299.2.9 loadu()	774
16.299.2.10 store()	774
16.299.2.11 maskstore()	774
16.299.2.12 storeu()	774
16.299.2.13 stream()	774
16.299.2.14 sll()	774
16.299.2.15 srl()	775
16.299.2.16 sra()	775
16.299.2.17 shuffle()	775
16.299.2.18 unpacklo_intrinsic()	775
16.299.2.19 unpackhi_intrinsic()	775
16.299.2.20 unpacklo()	775
16.299.2.21 unpackhi()	775
16.299.2.22 unpacklohi()	775
16.299.2.23 pack_even()	775
16.299.2.24 pack_odd()	776

16.299.2.25 pack()	776
16.299.2.26 transpose()	776
16.299.2.27 blend()	776
16.299.2.28 add()	776
16.299.2.29 addin()	776
16.299.2.30 sub()	776
16.299.2.31 subin()	777
16.299.2.32 mullo()	777
16.299.2.33 mul()	777
16.299.2.34 mulhi()	777
16.299.2.35 mulx()	777
16.299.2.36 fmadd()	777
16.299.2.37 fmaddin()	777
16.299.2.38 fmaddx()	777
16.299.2.39 fmaddxin()	778
16.299.2.40 fnmadd()	778
16.299.2.41 fnmaddin()	778
16.299.2.42 fnmaddx()	778
16.299.2.43 fnmaddxin()	778
16.299.2.44 fmsub()	778
16.299.2.45 fmsubin()	778
16.299.2.46 fmsubx()	778
16.299.2.47 fmsubxin()	779
16.299.2.48 eq()	779
16.299.2.49 greater()	779
16.299.2.50 lesser()	779
16.299.2.51 greater_eq()	779
16.299.2.52 lesser_eq()	779
16.299.2.53 hadd_to_scal()	779
16.299.2.54 round()	779
16.299.2.55 mask_high()	780
16.299.2.56 mulhi_fast()	780
16.299.2.57 mod()	780
16.299.2.58 signbits()	780
16.299.2.59 zero()	780
16.299.2.60 vor()	780
16.299.2.61 vxor()	780
16.299.2.62 vand()	780
16.299.2.63 vandnot()	781
16.299.3 Field Documentation	781
16.299.3.1 vect_size	781
16.299.3.2 alignment	781

16.300 Simd512i_base Struct Reference . . . . .	781
16.300.1 Member Typedef Documentation . . . . .	781
16.300.1.1 vect_t . . . . .	781
16.300.2 Member Function Documentation . . . . .	781
16.300.2.1 zero() . . . . .	782
16.300.2.2 vor() . . . . .	782
16.300.2.3 vxor() . . . . .	782
16.300.2.4 vand() . . . . .	782
16.300.2.5 vandnot() . . . . .	782
16.301 SimdChooser< T, bool, bool > Struct Template Reference . . . . .	782
16.302 SimdChooser< T, false, b > Struct Template Reference . . . . .	782
16.302.1 Member Typedef Documentation . . . . .	782
16.302.1.1 value . . . . .	782
16.303 SimdChooser< T, true, false > Struct Template Reference . . . . .	783
16.303.1 Member Typedef Documentation . . . . .	783
16.303.1.1 value . . . . .	783
16.304 SimdChooser< T, true, true > Struct Template Reference . . . . .	783
16.304.1 Member Typedef Documentation . . . . .	783
16.304.1.1 value . . . . .	783
16.305 simdToType< T > Struct Template Reference . . . . .	783
16.306 Single Struct Reference . . . . .	783
16.307 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference . . . . .	783
16.308 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference . . . . .	784
16.308.1 Member Typedef Documentation . . . . .	784
16.308.1.1 Field . . . . .	784
16.308.2 Field Documentation . . . . .	784
16.308.2.1 col . . . . .	784
16.308.2.2 row . . . . .	784
16.308.2.3 dat . . . . .	784
16.308.2.4 delayed . . . . .	785
16.308.2.5 kmax . . . . .	785
16.308.2.6 m . . . . .	785
16.308.2.7 n . . . . .	785
16.308.2.8 nnz . . . . .	785
16.308.2.9 nElements . . . . .	785
16.308.2.10 maxrow . . . . .	785
16.309 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference . . . . .	785
16.309.1 Member Typedef Documentation . . . . .	786
16.309.1.1 Field . . . . .	786
16.309.2 Field Documentation . . . . .	786
16.309.2.1 cst . . . . .	786
16.309.2.2 col . . . . .	786

16.309.2.3 row	786
16.309.2.4 dat	786
16.309.2.5 delayed	786
16.309.2.6 kmax	786
16.309.2.7 m	786
16.309.2.8 n	786
16.309.2.9 nnz	787
16.309.2.10 nElements	787
16.309.2.11 maxrow	787
16.310 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference	787
16.310.1 Member Typedef Documentation	787
16.310.1.1 Field	787
16.310.2 Field Documentation	788
16.310.2.1 delayed	788
16.310.2.2 kmax	788
16.310.2.3 m	788
16.310.2.4 n	788
16.310.2.5 nnz	788
16.310.2.6 nElements	788
16.310.2.7 maxrow	788
16.310.2.8 col	788
16.310.2.9 st	788
16.310.2.10 stend	788
16.310.2.11 dat	788
16.311 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference	789
16.311.1 Member Typedef Documentation	789
16.311.1.1 Field	789
16.311.2 Field Documentation	789
16.311.2.1 delayed	789
16.311.2.2 col	789
16.311.2.3 st	789
16.311.2.4 dat	789
16.311.2.5 kmax	790
16.311.2.6 m	790
16.311.2.7 n	790
16.311.2.8 nnz	790
16.311.2.9 nElements	790
16.311.2.10 maxrow	790
16.311.2.11 nOnes	790
16.311.2.12 nMOnes	790
16.311.2.13 nOthers	790
16.312 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference	790

16.312.1 Member Typedef Documentation	791
16.312.1.1 Field	791
16.312.2 Field Documentation	791
16.312.2.1 cst	791
16.312.2.2 delayed	791
16.312.2.3 kmax	791
16.312.2.4 m	791
16.312.2.5 n	791
16.312.2.6 nnz	791
16.312.2.7 nElements	792
16.312.2.8 maxrow	792
16.312.2.9 col	792
16.312.2.10 st	792
16.312.2.11 stend	792
16.312.2.12 dat	792
16.313 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference	792
16.313.1 Member Typedef Documentation	793
16.313.1.1 Field	793
16.313.2 Field Documentation	793
16.313.2.1 delayed	793
16.313.2.2 kmax	793
16.313.2.3 m	793
16.313.2.4 n	793
16.313.2.5 ld	793
16.313.2.6 nnz	793
16.313.2.7 nElements	793
16.313.2.8 maxrow	793
16.313.2.9 col	793
16.313.2.10 dat	794
16.314 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference	794
16.314.1 Field Documentation	794
16.314.1.1 delayed	794
16.314.1.2 chunk	794
16.314.1.3 m	794
16.314.1.4 n	794
16.314.1.5 ld	795
16.314.1.6 kmax	795
16.314.1.7 nnz	795
16.314.1.8 nElements	795
16.314.1.9 maxrow	795
16.314.1.10 nChunks	795
16.314.1.11 col	795

16.314.1.12 dat . . . . .	795
16.315 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference . . . . .	795
16.315.1 Field Documentation . . . . .	796
16.315.1.1 cst . . . . .	796
16.315.1.2 delayed . . . . .	796
16.315.1.3 chunk . . . . .	796
16.315.1.4 m . . . . .	796
16.315.1.5 n . . . . .	796
16.315.1.6 ld . . . . .	796
16.315.1.7 kmax . . . . .	796
16.315.1.8 nnz . . . . .	796
16.315.1.9 nElements . . . . .	797
16.315.1.10 maxrow . . . . .	797
16.315.1.11 nChunks . . . . .	797
16.315.1.12 col . . . . .	797
16.315.1.13 dat . . . . .	797
16.316 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference . . . . .	797
16.316.1 Member Typedef Documentation . . . . .	798
16.316.1.1 Field . . . . .	798
16.316.2 Field Documentation . . . . .	798
16.316.2.1 cst . . . . .	798
16.316.2.2 delayed . . . . .	798
16.316.2.3 kmax . . . . .	798
16.316.2.4 m . . . . .	798
16.316.2.5 n . . . . .	798
16.316.2.6 ld . . . . .	798
16.316.2.7 nnz . . . . .	798
16.316.2.8 nElements . . . . .	798
16.316.2.9 maxrow . . . . .	798
16.316.2.10 col . . . . .	798
16.316.2.11 dat . . . . .	799
16.317 Sparse< _Field, SparseMatrix_t::HYB_ZO > Struct Template Reference . . . . .	799
16.317.1 Member Typedef Documentation . . . . .	799
16.317.1.1 Field . . . . .	799
16.317.1.2 Self_t . . . . .	799
16.317.2 Field Documentation . . . . .	799
16.317.2.1 delayed . . . . .	799
16.317.2.2 kmax . . . . .	799
16.317.2.3 m . . . . .	800
16.317.2.4 n . . . . .	800
16.317.2.5 nnz . . . . .	800
16.317.2.6 maxrow . . . . .	800

16.317.2.7 nElements . . . . .	800
16.317.2.8 dat . . . . .	800
16.317.2.9 one . . . . .	800
16.317.2.10 mone . . . . .	800
16.318 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference . . . . .	800
16.318.1 Member Typedef Documentation . . . . .	801
16.318.1.1 Field . . . . .	801
16.318.2 Field Documentation . . . . .	801
16.318.2.1 delayed . . . . .	801
16.318.2.2 chunk . . . . .	801
16.318.2.3 kmax . . . . .	801
16.318.2.4 m . . . . .	801
16.318.2.5 n . . . . .	801
16.318.2.6 maxrow . . . . .	801
16.318.2.7 sigma . . . . .	802
16.318.2.8 nChunks . . . . .	802
16.318.2.9 nnz . . . . .	802
16.318.2.10 nElements . . . . .	802
16.318.2.11 perm . . . . .	802
16.318.2.12 st . . . . .	802
16.318.2.13 chunkSize . . . . .	802
16.318.2.14 col . . . . .	802
16.318.2.15 dat . . . . .	802
16.319 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference . . . . .	802
16.319.1 Member Typedef Documentation . . . . .	803
16.319.1.1 Field . . . . .	803
16.319.2 Field Documentation . . . . .	803
16.319.2.1 cst . . . . .	803
16.319.2.2 delayed . . . . .	803
16.319.2.3 chunk . . . . .	803
16.319.2.4 kmax . . . . .	803
16.319.2.5 m . . . . .	803
16.319.2.6 n . . . . .	804
16.319.2.7 maxrow . . . . .	804
16.319.2.8 sigma . . . . .	804
16.319.2.9 nChunks . . . . .	804
16.319.2.10 nnz . . . . .	804
16.319.2.11 nElements . . . . .	804
16.319.2.12 perm . . . . .	804
16.319.2.13 st . . . . .	804
16.319.2.14 chunkSize . . . . .	804
16.319.2.15 col . . . . .	804

16.319.2.16 dat . . . . .	804
16.320 SpMat< Field, flag > Struct Template Reference . . . . .	804
16.320.1 Field Documentation . . . . .	805
16.320.1.1 _coo . . . . .	805
16.320.1.2 _csr . . . . .	805
16.320.1.3 _ell . . . . .	805
16.321 StatsMatrix Struct Reference . . . . .	805
16.321.1 Field Documentation . . . . .	806
16.321.1.1 rowdim . . . . .	806
16.321.1.2 coldim . . . . .	806
16.321.1.3 nOnes . . . . .	806
16.321.1.4 nMOnes . . . . .	806
16.321.1.5 nOthers . . . . .	806
16.321.1.6 nnz . . . . .	806
16.321.1.7 maxRow . . . . .	806
16.321.1.8 minRow . . . . .	806
16.321.1.9 averageRow . . . . .	806
16.321.1.10 deviationRow . . . . .	806
16.321.1.11 maxCol . . . . .	807
16.321.1.12 minCol . . . . .	807
16.321.1.13 averageCol . . . . .	807
16.321.1.14 deviationCol . . . . .	807
16.321.1.15 minColDifference . . . . .	807
16.321.1.16 maxColDifference . . . . .	807
16.321.1.17 averageColDifference . . . . .	807
16.321.1.18 deviationColDifference . . . . .	807
16.321.1.19 minRowDifference . . . . .	807
16.321.1.20 maxRowDifference . . . . .	807
16.321.1.21 averageRowDifference . . . . .	807
16.321.1.22 deviationRowDifference . . . . .	807
16.321.1.23 nDenseRows . . . . .	808
16.321.1.24 nDenseCols . . . . .	808
16.321.1.25 nEmptyRows . . . . .	808
16.321.1.26 nEmptyCols . . . . .	808
16.321.1.27 nEmptyColsEnd . . . . .	808
16.321.1.28 denseRows . . . . .	808
16.321.1.29 denseCols . . . . .	808
16.322 support_fast_mod< T > Struct Template Reference . . . . .	808
16.323 support_fast_mod< double > Struct Reference . . . . .	808
16.324 support_fast_mod< float > Struct Reference . . . . .	809
16.325 support_fast_mod< int64_t > Struct Reference . . . . .	809
16.326 support_simd< T > Struct Template Reference . . . . .	809

16.327 support_simd_add< T > Struct Template Reference . . . . .	810
16.328 support_simd_mod< T > Struct Template Reference . . . . .	810
16.329 Test< Elt > Class Template Reference . . . . .	810
16.329.1 Member Typedef Documentation . . . . .	811
16.329.1.1 Field . . . . .	811
16.329.1.2 Elt_ptr . . . . .	811
16.329.1.3 Residu . . . . .	811
16.329.1.4 enable_if_t . . . . .	811
16.329.1.5 is_same_element . . . . .	811
16.329.1.6 enable_if_no_simd_t . . . . .	811
16.329.1.7 enable_if_simd128_t . . . . .	811
16.329.1.8 enable_if_simd256_t . . . . .	812
16.329.1.9 enable_if_simd512_t . . . . .	812
16.329.2 Constructor & Destructor Documentation . . . . .	812
16.329.2.1 Test() . . . . .	812
16.329.3 Member Function Documentation . . . . .	812
16.329.3.1 cardinality() [1/2] . . . . .	812
16.329.3.2 cardinality() [2/2] . . . . .	812
16.329.3.3 test_ftranspose() . . . . .	812
16.329.3.4 doTests() . . . . .	812
16.329.3.5 run() . . . . .	812
16.329.4 Field Documentation . . . . .	812
16.329.4.1 F . . . . .	813
16.329.4.2 _mm . . . . .	813
16.329.4.3 _nn . . . . .	813
16.330 TestOneMethod< Simd > Class Template Reference . . . . .	813
16.330.1 Member Typedef Documentation . . . . .	814
16.330.1.1 Element . . . . .	814
16.330.1.2 vect_t . . . . .	814
16.330.1.3 vectElt . . . . .	814
16.330.1.4 enable_if_t . . . . .	814
16.330.2 Constructor & Destructor Documentation . . . . .	814
16.330.2.1 TestOneMethod() . . . . .	814
16.330.3 Member Function Documentation . . . . .	814
16.330.3.1 evaluate_scalar_method() [1/3] . . . . .	814
16.330.3.2 evaluate_scalar_method() [2/3] . . . . .	814
16.330.3.3 evaluate_scalar_method() [3/3] . . . . .	815
16.330.3.4 evaluate_simd_method() [1/2] . . . . .	815
16.330.3.5 evaluate_simd_method() [2/2] . . . . .	815
16.330.3.6 getStatus() . . . . .	815
16.330.3.7 getTestName() . . . . .	815
16.330.3.8 writeResultLine() . . . . .	815

16.330.3.9 writeDebugData()	815
16.330.4 Field Documentation	815
16.330.4.1 vect_size	815
16.330.4.2 nb_lref	815
16.330.4.3 name	815
16.330.4.4 inputs	816
16.330.4.5 outputs_simd	816
16.330.4.6 outputs_scalar	816
16.331 tfn_minus Struct Reference	816
16.331.1 Member Function Documentation	816
16.331.1.1 operator>()	816
16.332 tfn_minus_eq Struct Reference	816
16.332.1 Member Function Documentation	816
16.332.1.1 operator>()	816
16.333 tfn_mul Struct Reference	817
16.333.1 Member Function Documentation	817
16.333.1.1 operator>()	817
16.334 tfn_mul_eq Struct Reference	817
16.334.1 Member Function Documentation	817
16.334.1.1 operator>()	817
16.335 tfn_plus Struct Reference	817
16.335.1 Member Function Documentation	817
16.335.1.1 operator>()	818
16.336 tfn_plus_eq Struct Reference	818
16.336.1 Member Function Documentation	818
16.336.1.1 operator>()	818
16.337 Threads Struct Reference	818
16.338 ThreeD Struct Reference	818
16.339 ThreeDAdaptive Struct Reference	818
16.340 ThreeDInPlace Struct Reference	818
16.341 TRSMHelper< RectIterTrait, ParSeqTrait > Struct Template Reference	819
16.341.1 Detailed Description	819
16.341.2 Constructor & Destructor Documentation	819
16.341.2.1 TRSMHelper() [1/3]	819
16.341.2.2 TRSMHelper() [2/3]	819
16.341.2.3 TRSMHelper() [3/3]	819
16.341.3 Member Function Documentation	819
16.341.3.1 pMMH() [1/2]	820
16.341.3.2 pMMH() [2/2]	820
16.341.4 Field Documentation	820
16.341.4.1 parseq	820
16.342 TwoD Struct Reference	820

16.343 TwoDAdaptive Struct Reference . . . . .	820
16.344 UnparametricTag Struct Reference . . . . .	820
16.344.1 Detailed Description . . . . .	820
16.345 width< T > Struct Template Reference . . . . .	820
16.345.1 Field Documentation . . . . .	821
16.345.1.1 value . . . . .	821
16.346 width< double > Struct Reference . . . . .	821
16.346.1 Field Documentation . . . . .	821
16.346.1.1 value . . . . .	821
16.347 width< float > Struct Reference . . . . .	821
16.347.1 Field Documentation . . . . .	821
16.347.1.1 value . . . . .	821
16.348 Winograd Struct Reference . . . . .	821
16.349 WinogradPar Struct Reference . . . . .	821
<b>17 File Documentation . . . . .</b>	<b>823</b>
17.1 101-fgemv.C File Reference . . . . .	823
17.1.1 Function Documentation . . . . .	823
17.1.1.1 main() . . . . .	823
17.2 2x2-fgemv.C File Reference . . . . .	823
17.2.1 Function Documentation . . . . .	823
17.2.1.1 main() . . . . .	824
17.3 2x2-ftsrv.C File Reference . . . . .	824
17.3.1 Function Documentation . . . . .	824
17.3.1.1 main() . . . . .	824
17.4 2x2-pluq.C File Reference . . . . .	824
17.4.1 Function Documentation . . . . .	824
17.4.1.1 main() . . . . .	824
17.5 align-allocator.h File Reference . . . . .	825
17.6 args-parser.h File Reference . . . . .	825
17.6.1 Macro Definition Documentation . . . . .	825
17.6.1.1 TYPE_BOOL . . . . .	826
17.6.1.2 END_OF_ARGUMENTS . . . . .	826
17.6.1.3 type_integer . . . . .	826
17.6.2 Enumeration Type Documentation . . . . .	826
17.6.2.1 ArgumentType . . . . .	826
17.6.3 Function Documentation . . . . .	826
17.6.3.1 printHelpMessage() . . . . .	826
17.6.3.2 findArgument() . . . . .	826
17.6.3.3 getListArgs() . . . . .	826
17.7 arithprog.C File Reference . . . . .	827
17.7.1 Macro Definition Documentation . . . . .	827

17.7.1.1 CUBE	827
17.7.1.2 GFOPS	827
17.7.2 Typedef Documentation	827
17.7.2.1 TTimer	828
17.7.3 Function Documentation	828
17.7.3.1 main()	828
17.8 benchmark-charpoly-mp.C File Reference	828
17.8.1 Macro Definition Documentation	828
17.8.1.1 __FFLASFFPACK_FORCE_SEQ	828
17.8.2 Function Documentation	828
17.8.2.1 main()	828
17.9 benchmark-charpoly.C File Reference	828
17.9.1 Macro Definition Documentation	829
17.9.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	829
17.9.2 Function Documentation	829
17.9.2.1 run_with_field()	829
17.9.2.2 main()	829
17.10 benchmark-checkers.C File Reference	829
17.10.1 Macro Definition Documentation	830
17.10.1.1 ENABLE_ALL_CHECKINGS	830
17.10.1.2 _NR_TESTS	830
17.10.1.3 _MAX_SIZE_MATRICES	830
17.10.1.4 CUBE	830
17.10.2 Function Documentation	830
17.10.2.1 main()	830
17.11 benchmark-dgemm.C File Reference	830
17.11.1 Macro Definition Documentation	831
17.11.1.1 CBLAS_GEMM	831
17.11.2 Typedef Documentation	831
17.11.2.1 TTimer	831
17.11.2.2 Floats	831
17.11.3 Function Documentation	831
17.11.3.1 main()	831
17.12 benchmark-dgetrf.C File Reference	831
17.12.1 Macro Definition Documentation	832
17.12.1.1 __FFLASFFPACK_HAVE_DGETRF	832
17.12.2 Typedef Documentation	832
17.12.2.1 TTimer	832
17.12.3 Function Documentation	832
17.12.3.1 main()	832
17.13 benchmark-dgetri.C File Reference	832
17.13.1 Typedef Documentation	833

17.13.1.1 TTimer . . . . .	833
17.13.2 Function Documentation . . . . .	833
17.13.2.1 main() . . . . .	833
17.14 benchmark-dsytrf.C File Reference . . . . .	833
17.14.1 Macro Definition Documentation . . . . .	833
17.14.1.1 EFGFF . . . . .	833
17.14.2 Typedef Documentation . . . . .	834
17.14.2.1 TTimer . . . . .	834
17.14.3 Function Documentation . . . . .	834
17.14.3.1 main() . . . . .	834
17.15 benchmark-dtrsm.C File Reference . . . . .	834
17.15.1 Typedef Documentation . . . . .	834
17.15.1.1 TTimer . . . . .	834
17.15.2 Function Documentation . . . . .	834
17.15.2.1 main() . . . . .	834
17.16 benchmark-dtrtri.C File Reference . . . . .	835
17.16.1 Macro Definition Documentation . . . . .	835
17.16.1.1 __FLLASFFPACK_HAVE_DTRTRI . . . . .	835
17.16.2 Typedef Documentation . . . . .	835
17.16.2.1 TTimer . . . . .	835
17.16.3 Function Documentation . . . . .	835
17.16.3.1 main() . . . . .	835
17.17 benchmark-fadd-lvl2.C File Reference . . . . .	835
17.17.1 Macro Definition Documentation . . . . .	836
17.17.1.1 __FLLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	836
17.17.2 Function Documentation . . . . .	836
17.17.2.1 main() . . . . .	836
17.18 benchmark-fdot.C File Reference . . . . .	836
17.18.1 Macro Definition Documentation . . . . .	836
17.18.1.1 __FLLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	836
17.18.2 Function Documentation . . . . .	837
17.18.2.1 run_with_field() . . . . .	837
17.18.2.2 main() . . . . .	837
17.19 benchmark-fgemm-mp.C File Reference . . . . .	837
17.19.1 Macro Definition Documentation . . . . .	837
17.19.1.1 __FLLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	837
17.19.2 Function Documentation . . . . .	837
17.19.2.1 tmain() . . . . .	838
17.19.2.2 main() . . . . .	838
17.20 benchmark-fgemm-rns.C File Reference . . . . .	838
17.20.1 Macro Definition Documentation . . . . .	838
17.20.1.1 __FLLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	838

17.20.2 Typedef Documentation	838
17.20.2.1 RNS	838
17.20.2.2 Field	839
17.20.2.3 Element_ptr	839
17.20.2.4 ConstElement_ptr	839
17.20.2.5 THREADS	839
17.20.2.6 GRAIN	839
17.20.2.7 TWOD	839
17.20.2.8 TWODA	839
17.20.2.9 THREED	839
17.20.2.10 THREEDA	839
17.20.2.11 THREEDIP	839
17.20.2.12 PSeq	839
17.20.3 Function Documentation	839
17.20.3.1 main()	840
17.21 benchmark-fgemm.C File Reference	840
17.21.1 Macro Definition Documentation	840
17.21.1.1 CLASSIC_HYBRID	840
17.21.2 Function Documentation	840
17.21.2.1 main()	840
17.22 benchmark-fgemv-mp.C File Reference	840
17.22.1 Macro Definition Documentation	841
17.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	841
17.22.2 Function Documentation	841
17.22.2.1 write_matrix()	841
17.23 benchmark-fgemv.C File Reference	841
17.23.1 Macro Definition Documentation	842
17.23.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	842
17.23.2 Function Documentation	842
17.23.2.1 fill_value()	842
17.23.2.2 genData()	842
17.23.2.3 check_result()	843
17.23.2.4 benchmark_with_timer()	843
17.23.2.5 benchmark_disp()	843
17.23.2.6 benchmark_in_Field()	844
17.23.2.7 benchmark_with_field() [1/2]	844
17.23.2.8 benchmark_with_field() [2/2]	844
17.23.2.9 main()	844
17.24 benchmark-fgesv.C File Reference	844
17.24.1 Macro Definition Documentation	845
17.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	845
17.24.2 Function Documentation	845

17.24.2.1 main()	845
17.25 benchmark-fsyr2k.C File Reference	845
17.25.1 Function Documentation	845
17.25.1.1 main()	845
17.26 benchmark-fsyrrk.C File Reference	846
17.26.1 Macro Definition Documentation	846
17.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	846
17.26.2 Function Documentation	846
17.26.2.1 main()	846
17.27 benchmark-fsytrf.C File Reference	846
17.27.1 Macro Definition Documentation	847
17.27.1.1 __FFPACK_FSYTRF_BC_CROUT	847
17.27.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	847
17.27.1.3 CUBE	847
17.27.2 Function Documentation	847
17.27.2.1 main()	847
17.28 benchmark-ftsrm-mp.C File Reference	847
17.28.1 Macro Definition Documentation	847
17.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	847
17.28.2 Function Documentation	848
17.28.2.1 main()	848
17.29 benchmark-ftsrm.C File Reference	848
17.29.1 Macro Definition Documentation	848
17.29.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	848
17.29.2 Function Documentation	848
17.29.2.1 main()	848
17.30 benchmark-ftsrv.C File Reference	848
17.30.1 Macro Definition Documentation	849
17.30.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	849
17.30.2 Function Documentation	849
17.30.2.1 main()	849
17.31 benchmark-ftsrti.C File Reference	849
17.31.1 Macro Definition Documentation	849
17.31.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	849
17.31.1.2 CUBE	849
17.31.2 Function Documentation	850
17.31.2.1 main()	850
17.32 benchmark-inverse.C File Reference	850
17.32.1 Macro Definition Documentation	850
17.32.1.1 CUBE	850
17.32.2 Function Documentation	850
17.32.2.1 main()	850

17.33 benchmark-lqup-mp.C File Reference	850
17.33.1 Function Documentation	851
17.33.1.1 main()	851
17.34 benchmark-lqup.C File Reference	851
17.34.1 Macro Definition Documentation	851
17.34.1.1 CUBE	851
17.34.2 Function Documentation	851
17.34.2.1 main()	851
17.35 benchmark-pluq.C File Reference	852
17.35.1 Macro Definition Documentation	852
17.35.1.1 __Fflasffpack_openblas_nt_already_set	852
17.35.1.2 CUBE	852
17.35.2 Typedef Documentation	852
17.35.2.1 Field	852
17.35.3 Function Documentation	852
17.35.3.1 verification_PLUQ()	853
17.35.3.2 Rec_initialize()	853
17.35.3.3 main()	853
17.36 benchmark-quasisep.C File Reference	853
17.36.1 Macro Definition Documentation	853
17.36.1.1 __Fflasffpack_openblas_nt_already_set	854
17.36.2 Function Documentation	854
17.36.2.1 run_with_field()	854
17.36.2.2 main()	854
17.37 benchmark-storage-transpose.C File Reference	854
17.37.1 Function Documentation	854
17.37.1.1 main()	854
17.38 benchmark-wino.C File Reference	855
17.38.1 Macro Definition Documentation	855
17.38.1.1 CUBE	855
17.38.2 Function Documentation	855
17.38.2.1 launch_wino()	855
17.38.2.2 main()	855
17.39 bit_manipulation.h File Reference	855
17.39.1 Macro Definition Documentation	856
17.39.1.1 __has_builtin	856
17.39.2 Function Documentation	856
17.39.2.1 clz() [1/2]	856
17.39.2.2 clz() [2/2]	856
17.39.2.3 ctz() [1/2]	856
17.39.2.4 ctz() [2/2]	856
17.40 blockcuts.inl File Reference	856

17.40.1 Macro Definition Documentation . . . . .	858
17.40.1.1 __FFLASFFPACK_fflas_blockcuts_INL . . . . .	858
17.40.1.2 __FFLASFFPACK_MINBLOCKCUTS . . . . .	858
17.41 cast.h File Reference . . . . .	858
17.42 cblas.C File Reference . . . . .	858
17.42.1 Macro Definition Documentation . . . . .	858
17.42.1.1 __FFLASFFPACK_CONFIGURATION . . . . .	858
17.42.1.2 __FFLASFFPACK_HAVE_CBLAS . . . . .	859
17.42.2 Function Documentation . . . . .	859
17.42.2.1 main() . . . . .	859
17.43 charpoly.C File Reference . . . . .	859
17.43.1 Macro Definition Documentation . . . . .	859
17.43.1.1 CUBE . . . . .	859
17.43.1.2 GFOPS . . . . .	859
17.43.2 Typedef Documentation . . . . .	859
17.43.2.1 TTimer . . . . .	860
17.43.3 Function Documentation . . . . .	860
17.43.3.1 main() . . . . .	860
17.44 charpoly.C File Reference . . . . .	860
17.44.1 Function Documentation . . . . .	860
17.44.1.1 main() . . . . .	860
17.45 checker_charpoly.inl File Reference . . . . .	860
17.45.1 Macro Definition Documentation . . . . .	861
17.45.1.1 __FFLASFFPACK_checker_charpoly_INL . . . . .	861
17.46 checker_det.inl File Reference . . . . .	861
17.46.1 Macro Definition Documentation . . . . .	861
17.46.1.1 __FFLASFFPACK_checker_det_INL . . . . .	861
17.47 checker_empty.h File Reference . . . . .	861
17.48 checker_fgemm.inl File Reference . . . . .	861
17.48.1 Macro Definition Documentation . . . . .	862
17.48.1.1 __FFLASFFPACK_checker_fgemm_INL . . . . .	862
17.49 checker_ftsm.inl File Reference . . . . .	862
17.49.1 Macro Definition Documentation . . . . .	862
17.49.1.1 __FFLASFFPACK_checker_ftsm_INL . . . . .	862
17.50 checker_invert.inl File Reference . . . . .	862
17.50.1 Macro Definition Documentation . . . . .	862
17.50.1.1 __FFLASFFPACK_checker_invert_INL . . . . .	862
17.51 checker_pluq.inl File Reference . . . . .	863
17.51.1 Macro Definition Documentation . . . . .	863
17.51.1.1 __FFLASFFPACK_checker_pluq_INL . . . . .	863
17.52 checkers.doxy File Reference . . . . .	863
17.53 checkers_fflas.h File Reference . . . . .	863

17.54	checkers_fflas.inl File Reference	864
17.54.1	Macro Definition Documentation	864
17.54.1.1	FFLASFFPACK_checkers_fflas_inl_H	864
17.55	checkers_ffpack.h File Reference	864
17.56	checkers_ffpack.inl File Reference	865
17.56.1	Macro Definition Documentation	865
17.56.1.1	FFLASFFPACK_checkers_ffpack_inl_H	865
17.57	clapack.C File Reference	865
17.57.1	Macro Definition Documentation	865
17.57.1.1	__FFLASFFPACK_CONFIGURATION	866
17.57.1.2	__FFLASFFPACK_HAVE_LAPACK	866
17.57.1.3	__FFLASFFPACK_HAVE_CLAPACK	866
17.57.2	Function Documentation	866
17.57.2.1	main()	866
17.58	config-blas.h File Reference	866
17.58.1	Macro Definition Documentation	867
17.58.1.1	CBLAS_INT	867
17.58.1.2	CBLAS_ENUM_DEFINED_H	867
17.58.1.3	CBLAS_EXTERNALS	867
17.58.1.4	blas_enum	867
17.58.2	Enumeration Type Documentation	867
17.58.2.1	CBLAS_ORDER	867
17.58.2.2	CBLAS_TRANSPOSE	868
17.58.2.3	CBLAS_UPLO	868
17.58.2.4	CBLAS_DIAG	868
17.58.2.5	CBLAS_SIDE	868
17.58.3	Function Documentation	868
17.58.3.1	daxpy_()	868
17.58.3.2	saxpy_()	869
17.58.3.3	ddot_()	869
17.58.3.4	sdot_()	869
17.58.3.5	dasum_()	869
17.58.3.6	idamax_()	869
17.58.3.7	dnrm2_()	869
17.58.3.8	dgemv_()	870
17.58.3.9	sgemv_()	870
17.58.3.10	dger_()	870
17.58.3.11	sger_()	870
17.58.3.12	dcopy_()	871
17.58.3.13	scopy_()	871
17.58.3.14	dscal_()	871
17.58.3.15	sscal_()	871

17.58.3.16 dtrsm_()	871
17.58.3.17 strsm_()	871
17.58.3.18 dtrmm_()	872
17.58.3.19 strmm_()	872
17.58.3.20 sgemm_()	872
17.58.3.21 dgemm_()	873
17.58.3.22 cblas_dsyrk()	873
17.59 config.h File Reference	873
17.59.1 Macro Definition Documentation	874
17.59.1.1 HAVE_BLAS	874
17.59.1.2 HAVE_CBLAS	874
17.59.1.3 HAVE_CXX11	874
17.59.1.4 HAVE_DLFCN_H	874
17.59.1.5 HAVE_FLOAT_H	874
17.59.1.6 HAVE_INT128	874
17.59.1.7 HAVE_INTTYPES_H	875
17.59.1.8 HAVE_LAPACK	875
17.59.1.9 HAVE_LIMITS_H	875
17.59.1.10 HAVE_LITTLE_ENDIAN	875
17.59.1.11 HAVE_MEMORY_H	875
17.59.1.12 HAVE_PTHREAD_H	875
17.59.1.13 HAVE_STDDEF_H	875
17.59.1.14 HAVE_STDINT_H	875
17.59.1.15 HAVE_STDLIB_H	875
17.59.1.16 HAVE_STRINGS_H	875
17.59.1.17 HAVE_STRING_H	875
17.59.1.18 HAVE_SYS_STAT_H	875
17.59.1.19 HAVE_SYS_TIME_H	876
17.59.1.20 HAVE_SYS_TYPES_H	876
17.59.1.21 HAVE_UNISTD_H	876
17.59.1.22 LT_OBJDIR	876
17.59.1.23 OPENBLAS_NUM_THREADS	876
17.59.1.24 PACKAGE	876
17.59.1.25 PACKAGE_BUGREPORT	876
17.59.1.26 PACKAGE_NAME	876
17.59.1.27 PACKAGE_STRING	876
17.59.1.28 PACKAGE_TARNAME	876
17.59.1.29 PACKAGE_URL	876
17.59.1.30 PACKAGE_VERSION	876
17.59.1.31 SIZEOF_CHAR	877
17.59.1.32 SIZEOF_INT	877
17.59.1.33 SIZEOF_LONG	877

17.59.1.34	SIZEOF_LONG_LONG	877
17.59.1.35	SIZEOF_SHORT	877
17.59.1.36	SIZEOF__INT64_T	877
17.59.1.37	STDC_HEADERS	877
17.59.1.38	USE_OPENMP	877
17.59.1.39	VERSION	877
17.60	config.h File Reference	877
17.60.1	Macro Definition Documentation	878
17.60.1.1	__FFLASFFPACK_HAVE_BLAS	878
17.60.1.2	__FFLASFFPACK_HAVE_CBLAS	878
17.60.1.3	__FFLASFFPACK_HAVE_CXX11	878
17.60.1.4	__FFLASFFPACK_HAVE_DLFCN_H	878
17.60.1.5	__FFLASFFPACK_HAVE_FLOAT_H	878
17.60.1.6	__FFLASFFPACK_HAVE_INT128	879
17.60.1.7	__FFLASFFPACK_HAVE_INTTYPES_H	879
17.60.1.8	__FFLASFFPACK_HAVE_LAPACK	879
17.60.1.9	__FFLASFFPACK_HAVE_LIMITS_H	879
17.60.1.10	__FFLASFFPACK_HAVE_LITTLE_ENDIAN	879
17.60.1.11	__FFLASFFPACK_HAVE_MEMORY_H	879
17.60.1.12	__FFLASFFPACK_HAVE_PTHREAD_H	879
17.60.1.13	__FFLASFFPACK_HAVE_STDDEF_H	879
17.60.1.14	__FFLASFFPACK_HAVE_STDINT_H	879
17.60.1.15	__FFLASFFPACK_HAVE_STDLIB_H	879
17.60.1.16	__FFLASFFPACK_HAVE_STRINGS_H	879
17.60.1.17	__FFLASFFPACK_HAVE_STRING_H	879
17.60.1.18	__FFLASFFPACK_HAVE_SYS_STAT_H	880
17.60.1.19	__FFLASFFPACK_HAVE_SYS_TIME_H	880
17.60.1.20	__FFLASFFPACK_HAVE_SYS_TYPES_H	880
17.60.1.21	__FFLASFFPACK_HAVE_UNISTD_H	880
17.60.1.22	__FFLASFFPACK_LT_OBJDIR	880
17.60.1.23	__FFLASFFPACK_OPENBLAS_NUM_THREADS	880
17.60.1.24	__FFLASFFPACK_PACKAGE	880
17.60.1.25	__FFLASFFPACK_PACKAGE_BUGREPORT	880
17.60.1.26	__FFLASFFPACK_PACKAGE_NAME	880
17.60.1.27	__FFLASFFPACK_PACKAGE_STRING	880
17.60.1.28	__FFLASFFPACK_PACKAGE_TARNAME	880
17.60.1.29	__FFLASFFPACK_PACKAGE_URL	880
17.60.1.30	__FFLASFFPACK_PACKAGE_VERSION	881
17.60.1.31	__FFLASFFPACK_SIZEOF_CHAR	881
17.60.1.32	__FFLASFFPACK_SIZEOF_INT	881
17.60.1.33	__FFLASFFPACK_SIZEOF_LONG	881
17.60.1.34	__FFLASFFPACK_SIZEOF_LONG_LONG	881

17.60.1.35	<a href="#">__FFLASFFPACK_SIZEOF_SHORT</a>	881
17.60.1.36	<a href="#">__FFLASFFPACK_SIZEOF__INT64_T</a>	881
17.60.1.37	<a href="#">__FFLASFFPACK_STDC_HEADERS</a>	881
17.60.1.38	<a href="#">__FFLASFFPACK_USE_OPENMP</a>	881
17.60.1.39	<a href="#">__FFLASFFPACK_VERSION</a>	881
17.61	<a href="#">coo.h File Reference</a>	881
17.62	<a href="#">coo_spmml.inl File Reference</a>	882
17.62.1	Macro Definition Documentation	883
17.62.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_spmml_INL</a>	883
17.63	<a href="#">coo_spmv.inl File Reference</a>	883
17.63.1	Macro Definition Documentation	884
17.63.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_spmv_INL</a>	884
17.64	<a href="#">coo_utils.inl File Reference</a>	884
17.64.1	Macro Definition Documentation	884
17.64.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_utils_INL</a>	884
17.65	<a href="#">csr.h File Reference</a>	884
17.66	<a href="#">csr_hyb.h File Reference</a>	885
17.67	<a href="#">csr_hyb_pspmm.inl File Reference</a>	885
17.67.1	Macro Definition Documentation	886
17.67.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL</a>	886
17.68	<a href="#">csr_hyb_pspmv.inl File Reference</a>	886
17.68.1	Macro Definition Documentation	886
17.68.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL</a>	886
17.69	<a href="#">csr_hyb_spmml.inl File Reference</a>	887
17.69.1	Macro Definition Documentation	887
17.69.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_spmml_INL</a>	887
17.70	<a href="#">csr_hyb_spmv.inl File Reference</a>	887
17.70.1	Macro Definition Documentation	888
17.70.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL</a>	888
17.71	<a href="#">csr_hyb_utils.inl File Reference</a>	888
17.71.1	Macro Definition Documentation	888
17.71.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL</a>	888
17.72	<a href="#">csr_pspmm.inl File Reference</a>	888
17.72.1	Macro Definition Documentation	889
17.72.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_pspmm_INL</a>	889
17.73	<a href="#">csr_pspmv.inl File Reference</a>	889
17.73.1	Macro Definition Documentation	890
17.73.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_pspmv_INL</a>	890
17.74	<a href="#">csr_spmml.inl File Reference</a>	890
17.74.1	Macro Definition Documentation	891
17.74.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_spmml_INL</a>	891
17.75	<a href="#">csr_spmv.inl File Reference</a>	891

17.75.1 Macro Definition Documentation	892
17.75.1.1 __FflasFFPACK_fflas_sparse_CSR_spmv_INL	892
17.76 csr_utils.inl File Reference	892
17.77 cuda.C File Reference	893
17.77.1 Function Documentation	893
17.77.1.1 main()	893
17.78 debug.h File Reference	893
17.78.1 Detailed Description	893
17.78.2 Macro Definition Documentation	894
17.78.2.1 FflasFFPACK_check	894
17.78.2.2 FflasFFPACK_abort	894
17.79 det.C File Reference	894
17.79.1 Function Documentation	894
17.79.1.1 main()	894
17.80 ell.h File Reference	894
17.81 ell_pspmm.inl File Reference	895
17.81.1 Macro Definition Documentation	896
17.81.1.1 __FflasFFPACK_fflas_sparse_ELL_pspmm_INL	896
17.82 ell_spmv.inl File Reference	896
17.82.1 Macro Definition Documentation	896
17.82.1.1 __FflasFFPACK_fflas_sparse_ELL_spmv_INL	896
17.83 ell_simd.h File Reference	897
17.84 ell_simd_pspmv.inl File Reference	897
17.84.1 Macro Definition Documentation	898
17.84.1.1 __FflasFFPACK_fflas_sparse_ELL_simd_pspmv_INL	898
17.85 ell_simd_spmv.inl File Reference	898
17.85.1 Macro Definition Documentation	899
17.85.1.1 __FflasFFPACK_fflas_sparse_ELL_simd_spmv_INL	899
17.86 ell_simd_utils.inl File Reference	899
17.86.1 Macro Definition Documentation	899
17.86.1.1 __FflasFFPACK_fflas_sparse_ELL_simd_utils_INL	899
17.87 ell_spm.inl File Reference	899
17.87.1 Macro Definition Documentation	900
17.87.1.1 __FflasFFPACK_fflas_sparse_ELL_spm_INL	900
17.88 ell_spmv.inl File Reference	900
17.88.1 Macro Definition Documentation	901
17.88.1.1 __FflasFFPACK_fflas_sparse_ELL_spmv_INL	901
17.89 ell_utils.inl File Reference	901
17.89.1 Macro Definition Documentation	902
17.89.1.1 __FflasFFPACK_fflas_sparse_ELL_utils_INL	902
17.90 fblas.C File Reference	902
17.90.1 Macro Definition Documentation	902

17.90.1.1 __FFLASFFPACK_CONFIGURATION . . . . .	902
17.90.2 Function Documentation . . . . .	902
17.90.2.1 dgemm_() . . . . .	902
17.90.2.2 main() . . . . .	903
17.91 fflas-101_1.C File Reference . . . . .	903
17.91.1 Function Documentation . . . . .	903
17.91.1.1 main() . . . . .	903
17.92 fflas-101_3.C File Reference . . . . .	903
17.92.1 Function Documentation . . . . .	903
17.92.1.1 main() . . . . .	903
17.93 fflas-ffpack-config.h File Reference . . . . .	904
17.93.1 Detailed Description . . . . .	904
17.93.2 Macro Definition Documentation . . . . .	904
17.93.2.1 GCC_VERSION . . . . .	904
17.94 fflas-ffpack-default-thresholds.h File Reference . . . . .	904
17.94.1 Macro Definition Documentation . . . . .	904
17.94.1.1 __FFLASFFPACK_WINOTHRESHOLD . . . . .	904
17.94.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT . . . . .	904
17.94.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL . . . . .	905
17.94.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT . . . . .	905
17.94.1.5 __FFLASFFPACK_PLUQ_THRESHOLD . . . . .	905
17.94.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD . . . . .	905
17.94.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD . . . . .	905
17.94.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD . . . . .	905
17.94.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD . . . . .	905
17.94.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD . . . . .	905
17.94.1.11 __FFLASFFPACK_FSYRK_THRESHOLD . . . . .	905
17.95 fflas-ffpack-thresholds.h File Reference . . . . .	905
17.96 fflas-ffpack.doxy File Reference . . . . .	905
17.97 fflas-ffpack.h File Reference . . . . .	905
17.97.1 Detailed Description . . . . .	905
17.98 fflas.doxy File Reference . . . . .	906
17.99 fflas.h File Reference . . . . .	906
17.99.1 Detailed Description . . . . .	907
17.99.2 Macro Definition Documentation . . . . .	907
17.99.2.1 WINOTHRESHOLD . . . . .	907
17.99.2.2 DOUBLE_TO_FLOAT_CROSSOVER . . . . .	907
17.100 fflas_101.C File Reference . . . . .	907
17.100.1 Function Documentation . . . . .	907
17.100.1.1 main() . . . . .	907
17.101 fflas_101_lvl1.C File Reference . . . . .	907
17.101.1 Function Documentation . . . . .	908

17.101.1.1 main()	908
17.102 fflas_bounds.inl File Reference	908
17.102.1 Macro Definition Documentation	908
17.102.1.1 __FFLASFFPACK_fflas_bounds_INL	908
17.102.1.2 FFLAS_INT_TYPE	909
17.103 fflas_c.h File Reference	909
17.103.1 Macro Definition Documentation	911
17.103.1.1 FFLAS_COMPILED	911
17.103.2 Enumeration Type Documentation	911
17.103.2.1 FFLAS_C_ORDER	911
17.103.2.2 FFLAS_C_TRANSPOSE	911
17.103.2.3 FFLAS_C_UPLO	911
17.103.2.4 FFLAS_C_DIAG	912
17.103.2.5 FFLAS_C_SIDE	912
17.103.2.6 FFLAS_C_BASE	912
17.103.3 Function Documentation	912
17.103.3.1 freducein_1_modular_double()	913
17.103.3.2 freduce_1_modular_double()	913
17.103.3.3 fnegin_1_modular_double()	913
17.103.3.4 fneg_1_modular_double()	913
17.103.3.5 fzero_1_modular_double()	913
17.103.3.6 fiszero_1_modular_double()	913
17.103.3.7 fequal_1_modular_double()	914
17.103.3.8 fassign_1_modular_double()	914
17.103.3.9 fscal_1_modular_double()	914
17.103.3.10 fscal_1_modular_double()	914
17.103.3.11 faxpy_1_modular_double()	914
17.103.3.12 fdot_1_modular_double()	915
17.103.3.13 fswap_1_modular_double()	915
17.103.3.14 fadd_1_modular_double()	915
17.103.3.15 fsub_1_modular_double()	915
17.103.3.16 faddin_1_modular_double()	916
17.103.3.17 fsubin_1_modular_double()	916
17.103.3.18 fassign_2_modular_double()	916
17.103.3.19 fzero_2_modular_double()	916
17.103.3.20 fequal_2_modular_double()	916
17.103.3.21 fiszero_2_modular_double()	917
17.103.3.22 fidentity_2_modular_double()	917
17.103.3.23 freducein_2_modular_double()	917
17.103.3.24 freduce_2_modular_double()	917
17.103.3.25 fnegin_2_modular_double()	917
17.103.3.26 fneg_2_modular_double()	918

17.103.3.27 fscaln_2_modular_double()	918
17.103.3.28 fscal_2_modular_double()	918
17.103.3.29 faxpy_2_modular_double()	918
17.103.3.30 fmove_2_modular_double()	918
17.103.3.31 fadd_2_modular_double()	919
17.103.3.32 fsub_2_modular_double()	919
17.103.3.33 fsubin_2_modular_double()	919
17.103.3.34 faddin_2_modular_double()	919
17.103.3.35 fgemv_2_modular_double()	920
17.103.3.36 fger_2_modular_double()	920
17.103.3.37 ftrsv_2_modular_double()	920
17.103.3.38 ftrsm_3_modular_double()	920
17.103.3.39 ftrmm_3_modular_double()	921
17.103.3.40 fgemm_3_modular_double()	921
17.103.3.41 fsquare_3_modular_double()	921
17.104 fflas_enum.h File Reference	922
17.105 fflas_fadd.h File Reference	922
17.106 fflas_fadd.inl File Reference	924
17.106.1 Macro Definition Documentation	925
17.106.1.1 __FFLASFFPACK_fadd_INL	925
17.107 fflas_fassign.h File Reference	925
17.108 fflas_fassign.inl File Reference	925
17.108.1 Macro Definition Documentation	926
17.108.1.1 __FFLASFFPACK_fassign_INL	926
17.109 fflas_faxpy.inl File Reference	926
17.109.1 Macro Definition Documentation	927
17.109.1.1 __FFLASFFPACK_faxpy_INL	927
17.110 fflas_fdot.inl File Reference	927
17.110.1 Macro Definition Documentation	928
17.110.1.1 __FFLASFFPACK_fdot_INL	928
17.111 fflas_fgemm.inl File Reference	928
17.111.1 Macro Definition Documentation	930
17.111.1.1 __FFLASFFPACK_fgemm_INL	930
17.112 fflas_fgemv.inl File Reference	930
17.112.1 Macro Definition Documentation	931
17.112.1.1 __FFLASFFPACK_fgemv_INL	931
17.113 fflas_fgemv_mp.inl File Reference	932
17.113.1 Macro Definition Documentation	932
17.113.1.1 __FFLASFFPACK_fgemv_mp_INL	932
17.114 fflas_fger.inl File Reference	932
17.114.1 Macro Definition Documentation	933
17.114.1.1 __FFLASFFPACK_fger_INL	933

17.115 fflas_fger_mp.inl File Reference . . . . .	934
17.115.1 Macro Definition Documentation . . . . .	934
17.115.1.1 __FFPACK_fger_mp_INL . . . . .	934
17.116 fflas_freduce.h File Reference . . . . .	934
17.117 fflas_freduce.inl File Reference . . . . .	935
17.117.1 Macro Definition Documentation . . . . .	937
17.117.1.1 __FFLASFFPACK_fflas_freduce_INL . . . . .	937
17.117.1.2 FFLASFFPACK_COPY_REDUCE . . . . .	937
17.118 fflas_freduce_mp.inl File Reference . . . . .	937
17.118.1 Macro Definition Documentation . . . . .	937
17.118.1.1 __FFLASFFPACK_fflas_freduce_mp_INL . . . . .	937
17.119 fflas_freivalds.inl File Reference . . . . .	937
17.119.1 Macro Definition Documentation . . . . .	938
17.119.1.1 __FFLASFFPACK_freivalds_INL . . . . .	938
17.120 fflas_fscal.h File Reference . . . . .	938
17.121 fflas_fscal.inl File Reference . . . . .	938
17.121.1 Macro Definition Documentation . . . . .	939
17.121.1.1 __FFLASFFPACK_fscal_INL . . . . .	940
17.122 fflas_fscal_mp.inl File Reference . . . . .	940
17.122.1 Macro Definition Documentation . . . . .	940
17.122.1.1 __FFLASFFPACK_fscal_mp_INL . . . . .	940
17.123 fflas_fsyr2k.inl File Reference . . . . .	940
17.123.1 Macro Definition Documentation . . . . .	941
17.123.1.1 __FFLASFFPACK_fflas_fsyr2k_INL . . . . .	941
17.124 fflas_fsyrk.inl File Reference . . . . .	941
17.124.1 Macro Definition Documentation . . . . .	943
17.124.1.1 __FFLASFFPACK_fflas_fsyrk_INL . . . . .	943
17.125 fflas_fsyrk_strassen.inl File Reference . . . . .	943
17.125.1 Macro Definition Documentation . . . . .	944
17.125.1.1 __FFLASFFPACK_fflas_fsyrk_strassen_INL . . . . .	944
17.126 fflas_ftrmm.inl File Reference . . . . .	944
17.126.1 Macro Definition Documentation . . . . .	944
17.126.1.1 __FFLASFFPACK_ftrmm_INL . . . . .	944
17.127 fflas_ftrsm.inl File Reference . . . . .	944
17.127.1 Macro Definition Documentation . . . . .	945
17.127.1.1 __FFLASFFPACK_ftrsm_INL . . . . .	945
17.128 fflas_ftrsm_mp.inl File Reference . . . . .	945
17.128.1 Detailed Description . . . . .	946
17.128.2 Macro Definition Documentation . . . . .	946
17.128.2.1 __FFPACK_ftrsm_mp_INL . . . . .	946
17.129 fflas_ftrsv.inl File Reference . . . . .	946
17.129.1 Macro Definition Documentation . . . . .	946

17.129.1.1 __FFLASFFPACK_ftsrv_INL . . . . .	946
17.130 fflas_helpers.inl File Reference . . . . .	946
17.130.1 Macro Definition Documentation . . . . .	947
17.130.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL . . . . .	948
17.131 fflas_intrinsic.h File Reference . . . . .	948
17.132 fflas_io.h File Reference . . . . .	948
17.133 fflas_L1_inst.C File Reference . . . . .	948
17.133.1 Macro Definition Documentation . . . . .	949
17.133.1.1 __FFLAS_L1_INST_C . . . . .	949
17.133.1.2 INST_OR_DECL . . . . .	949
17.133.1.3 FFLAS_FIELD [1/2] . . . . .	949
17.133.1.4 FFLAS_ELT [1/6] . . . . .	949
17.133.1.5 FFLAS_ELT [2/6] . . . . .	949
17.133.1.6 FFLAS_ELT [3/6] . . . . .	949
17.133.1.7 FFLAS_FIELD [2/2] . . . . .	949
17.133.1.8 FFLAS_ELT [4/6] . . . . .	949
17.133.1.9 FFLAS_ELT [5/6] . . . . .	950
17.133.1.10 FFLAS_ELT [6/6] . . . . .	950
17.134 fflas_L1_inst.h File Reference . . . . .	950
17.134.1 Macro Definition Documentation . . . . .	950
17.134.1.1 INST_OR_DECL . . . . .	950
17.134.1.2 FFLAS_FIELD [1/2] . . . . .	950
17.134.1.3 FFLAS_ELT [1/6] . . . . .	950
17.134.1.4 FFLAS_ELT [2/6] . . . . .	950
17.134.1.5 FFLAS_ELT [3/6] . . . . .	950
17.134.1.6 FFLAS_FIELD [2/2] . . . . .	951
17.134.1.7 FFLAS_ELT [4/6] . . . . .	951
17.134.1.8 FFLAS_ELT [5/6] . . . . .	951
17.134.1.9 FFLAS_ELT [6/6] . . . . .	951
17.135 fflas_L1_inst_impl.inl File Reference . . . . .	951
17.136 fflas_L2_inst.C File Reference . . . . .	952
17.136.1 Macro Definition Documentation . . . . .	952
17.136.1.1 __FFLAS_L2_INST_C . . . . .	953
17.136.1.2 INST_OR_DECL . . . . .	953
17.136.1.3 FFLAS_FIELD [1/2] . . . . .	953
17.136.1.4 FFLAS_ELT [1/6] . . . . .	953
17.136.1.5 FFLAS_ELT [2/6] . . . . .	953
17.136.1.6 FFLAS_ELT [3/6] . . . . .	953
17.136.1.7 FFLAS_FIELD [2/2] . . . . .	953
17.136.1.8 FFLAS_ELT [4/6] . . . . .	953
17.136.1.9 FFLAS_ELT [5/6] . . . . .	953
17.136.1.10 FFLAS_ELT [6/6] . . . . .	953

17.137 fflas_L2_inst.h File Reference . . . . .	953
17.137.1 Macro Definition Documentation . . . . .	954
17.137.1.1 INST_OR_DECL . . . . .	954
17.137.1.2 FFLAS_FIELD [1/2] . . . . .	954
17.137.1.3 FFLAS_ELT [1/6] . . . . .	954
17.137.1.4 FFLAS_ELT [2/6] . . . . .	954
17.137.1.5 FFLAS_ELT [3/6] . . . . .	954
17.137.1.6 FFLAS_FIELD [2/2] . . . . .	954
17.137.1.7 FFLAS_ELT [4/6] . . . . .	954
17.137.1.8 FFLAS_ELT [5/6] . . . . .	954
17.137.1.9 FFLAS_ELT [6/6] . . . . .	954
17.138 fflas_L2_inst_implem.inl File Reference . . . . .	955
17.139 fflas_L3_inst.C File Reference . . . . .	956
17.139.1 Macro Definition Documentation . . . . .	957
17.139.1.1 __FFLAS_L3_INST_C . . . . .	957
17.139.1.2 INST_OR_DECL . . . . .	957
17.139.1.3 FFLAS_FIELD [1/2] . . . . .	957
17.139.1.4 FFLAS_ELT [1/6] . . . . .	957
17.139.1.5 FFLAS_ELT [2/6] . . . . .	957
17.139.1.6 FFLAS_ELT [3/6] . . . . .	957
17.139.1.7 FFLAS_FIELD [2/2] . . . . .	957
17.139.1.8 FFLAS_ELT [4/6] . . . . .	957
17.139.1.9 FFLAS_ELT [5/6] . . . . .	957
17.139.1.10 FFLAS_ELT [6/6] . . . . .	957
17.140 fflas_L3_inst.h File Reference . . . . .	957
17.140.1 Macro Definition Documentation . . . . .	958
17.140.1.1 INST_OR_DECL . . . . .	958
17.140.1.2 FFLAS_FIELD [1/2] . . . . .	958
17.140.1.3 FFLAS_ELT [1/6] . . . . .	958
17.140.1.4 FFLAS_ELT [2/6] . . . . .	958
17.140.1.5 FFLAS_ELT [3/6] . . . . .	958
17.140.1.6 FFLAS_FIELD [2/2] . . . . .	958
17.140.1.7 FFLAS_ELT [4/6] . . . . .	958
17.140.1.8 FFLAS_ELT [5/6] . . . . .	958
17.140.1.9 FFLAS_ELT [6/6] . . . . .	959
17.141 fflas_L3_inst_implem.inl File Reference . . . . .	959
17.141.1 Macro Definition Documentation . . . . .	959
17.141.1.1 __FFLAS__TRSM_READONLY . . . . .	959
17.142 fflas_level1.inl File Reference . . . . .	960
17.142.1 Macro Definition Documentation . . . . .	962
17.142.1.1 __FFLASFFPACK_fflas_fflas_level1_INL . . . . .	962
17.143 fflas_level2.inl File Reference . . . . .	962

17.143.1 Macro Definition Documentation . . . . .	964
17.143.1.1 __FFLASFFPACK_fflas_fflas_level2_INL . . . . .	965
17.144 fflas_level3.inl File Reference . . . . .	965
17.144.1 Macro Definition Documentation . . . . .	967
17.144.1.1 __FFLASFFPACK_fflas_fflas_level3_INL . . . . .	967
17.144.1.2 __FFLAS__TRSM_READONLY . . . . .	967
17.145 fflas_lvl1.C File Reference . . . . .	967
17.145.1 Detailed Description . . . . .	968
17.145.2 Function Documentation . . . . .	968
17.145.2.1 freducein_1_modular_double() . . . . .	968
17.145.2.2 freduce_1_modular_double() . . . . .	968
17.145.2.3 fnegin_1_modular_double() . . . . .	969
17.145.2.4 fneg_1_modular_double() . . . . .	969
17.145.2.5 fzero_1_modular_double() . . . . .	969
17.145.2.6 fiszero_1_modular_double() . . . . .	969
17.145.2.7 fequal_1_modular_double() . . . . .	969
17.145.2.8 fassign_1_modular_double() . . . . .	969
17.145.2.9 fscaln_1_modular_double() . . . . .	970
17.145.2.10 fscal_1_modular_double() . . . . .	970
17.145.2.11 faxpy_1_modular_double() . . . . .	970
17.145.2.12 fdot_1_modular_double() . . . . .	970
17.145.2.13 fswap_1_modular_double() . . . . .	970
17.145.2.14 fadd_1_modular_double() . . . . .	971
17.145.2.15 fsub_1_modular_double() . . . . .	971
17.145.2.16 faddin_1_modular_double() . . . . .	971
17.145.2.17 fsubin_1_modular_double() . . . . .	971
17.146 fflas_lvl2.C File Reference . . . . .	972
17.146.1 Detailed Description . . . . .	973
17.146.2 Function Documentation . . . . .	973
17.146.2.1 fassign_2_modular_double() . . . . .	973
17.146.2.2 fzero_2_modular_double() . . . . .	973
17.146.2.3 fequal_2_modular_double() . . . . .	973
17.146.2.4 fiszero_2_modular_double() . . . . .	973
17.146.2.5 fidentity_2_modular_double() . . . . .	974
17.146.2.6 freducein_2_modular_double() . . . . .	974
17.146.2.7 freduce_2_modular_double() . . . . .	974
17.146.2.8 fnegin_2_modular_double() . . . . .	974
17.146.2.9 fneg_2_modular_double() . . . . .	974
17.146.2.10 fscaln_2_modular_double() . . . . .	975
17.146.2.11 fscal_2_modular_double() . . . . .	975
17.146.2.12 faxpy_2_modular_double() . . . . .	975
17.146.2.13 fmove_2_modular_double() . . . . .	975

17.146.2.14 fadd_2_modular_double()	976
17.146.2.15 fsub_2_modular_double()	976
17.146.2.16 fsubin_2_modular_double()	976
17.146.2.17 faddin_2_modular_double()	976
17.146.2.18 fgemv_2_modular_double()	976
17.146.2.19 fger_2_modular_double()	977
17.146.2.20 ftrsv_2_modular_double()	977
17.147 fflas_lvl3.C File Reference	977
17.147.1 Detailed Description	978
17.147.2 Function Documentation	978
17.147.2.1 ftrsm_3_modular_double()	978
17.147.2.2 ftrmm_3_modular_double()	978
17.147.2.3 fgemm_3_modular_double()	979
17.147.2.4 fsquare_3_modular_double()	979
17.148 fflas_memory.h File Reference	979
17.149 fflas_pfgemm.inl File Reference	980
17.149.1 Macro Definition Documentation	980
17.149.1.1 __FFLASFFPACK_fflas_pfgemm_INL	980
17.149.1.2 __FFLASFFPACK_SEQPARTHRESHOLD	980
17.149.1.3 __FFLASFFPACK_DIMKPENALTY	981
17.150 fflas_pftrsm.inl File Reference	981
17.150.1 Macro Definition Documentation	981
17.150.1.1 __FFLASFFPACK_fflas_pftrsm_INL	981
17.150.1.2 PTRSM_HYBRID_THRESHOLD	981
17.151 fflas_plevel1.h File Reference	981
17.152 fflas_randommatrix.h File Reference	982
17.153 fflas_simd.h File Reference	984
17.153.1 Macro Definition Documentation	985
17.153.1.1 SIMD_INT	985
17.153.1.2 INLINE	985
17.153.1.3 CONST	985
17.153.1.4 PURE	985
17.153.1.5 NORML_MOD	985
17.153.1.6 FLOAT_MOD	985
17.153.2 Typedef Documentation	986
17.153.2.1 Simd	986
17.154 fflas_sparse.C File Reference	986
17.154.1 Detailed Description	986
17.155 fflas_sparse.h File Reference	986
17.155.1 Macro Definition Documentation	990
17.155.1.1 index_t	990
17.155.1.2 ROUND_DOWN	990

17.155.1.3	<a href="#">__FFLASFFPACK_CACHE_LINE_SIZE</a>	990
17.155.1.4	<a href="#">assume_aligned</a>	990
17.155.1.5	<a href="#">DENSE_THRESHOLD</a>	991
17.156	<a href="#">fflas_sparse.inl</a> File Reference	991
17.156.1	Macro Definition Documentation	993
17.156.1.1	<a href="#">__FFLASFFPACK_fflas_fflas_sparse_INL</a>	993
17.157	<a href="#">fflas_transpose.h</a> File Reference	993
17.157.1	Detailed Description	993
17.157.2	Macro Definition Documentation	994
17.157.2.1	<a href="#">FFLAS_TRANSPOSE_BLOCKSIZE</a>	994
17.157.2.2	<a href="#">LD</a>	994
17.157.2.3	<a href="#">ST</a>	994
17.158	<a href="#">ffpack-fgesv.C</a> File Reference	994
17.158.1	Function Documentation	994
17.158.1.1	<a href="#">main()</a>	994
17.159	<a href="#">ffpack-solve.C</a> File Reference	994
17.159.1	Function Documentation	995
17.159.1.1	<a href="#">main()</a>	995
17.160	<a href="#">ffpack.C</a> File Reference	995
17.160.1	Detailed Description	998
17.160.2	Function Documentation	998
17.160.2.1	<a href="#">LAPACKPerm2MathPerm()</a>	998
17.160.2.2	<a href="#">MathPerm2LAPACKPerm()</a>	998
17.160.2.3	<a href="#">MatrixApplyS_modular_double()</a>	999
17.160.2.4	<a href="#">PermApplyS_double()</a>	999
17.160.2.5	<a href="#">MatrixApplyT_modular_double()</a>	999
17.160.2.6	<a href="#">PermApplyT_double()</a>	999
17.160.2.7	<a href="#">composePermutationsLLM()</a>	999
17.160.2.8	<a href="#">composePermutationsLLL()</a>	1000
17.160.2.9	<a href="#">composePermutationsMLM()</a>	1000
17.160.2.10	<a href="#">cyclic_shift_mathPerm()</a>	1000
17.160.2.11	<a href="#">cyclic_shift_row_modular_double()</a>	1000
17.160.2.12	<a href="#">cyclic_shift_col_modular_double()</a>	1000
17.160.2.13	<a href="#">applyP_modular_double()</a>	1000
17.160.2.14	<a href="#">fgetrsin_modular_double()</a>	1001
17.160.2.15	<a href="#">fgetrsv_modular_double()</a>	1001
17.160.2.16	<a href="#">fgesvin_modular_double()</a>	1001
17.160.2.17	<a href="#">fgesv_modular_double()</a>	1002
17.160.2.18	<a href="#">ftrtri_modular_double()</a>	1002
17.160.2.19	<a href="#">trinv_left_modular_double()</a>	1002
17.160.2.20	<a href="#">ftrtrm_modular_double()</a>	1002
17.160.2.21	<a href="#">PLUQ_modular_double()</a>	1003

17.160.2.22 LUdivine_modular_double()	1003
17.160.2.23 ColumnEchelonForm_modular_double()	1003
17.160.2.24 RowEchelonForm_modular_double()	1003
17.160.2.25 ReducedColumnEchelonForm_modular_double()	1004
17.160.2.26 ReducedRowEchelonForm_modular_double()	1004
17.160.2.27 ColumnEchelonForm_modular_float()	1004
17.160.2.28 RowEchelonForm_modular_float()	1004
17.160.2.29 ReducedColumnEchelonForm_modular_float()	1005
17.160.2.30 ReducedRowEchelonForm_modular_float()	1005
17.160.2.31 ColumnEchelonForm_modular_int32_t()	1005
17.160.2.32 RowEchelonForm_modular_int32_t()	1005
17.160.2.33 ReducedColumnEchelonForm_modular_int32_t()	1006
17.160.2.34 ReducedRowEchelonForm_modular_int32_t()	1006
17.160.2.35 pColumnEchelonForm_modular_double()	1006
17.160.2.36 pRowEchelonForm_modular_double()	1006
17.160.2.37 pReducedColumnEchelonForm_modular_double()	1007
17.160.2.38 pReducedRowEchelonForm_modular_double()	1007
17.160.2.39 pColumnEchelonForm_modular_float()	1007
17.160.2.40 pRowEchelonForm_modular_float()	1007
17.160.2.41 pReducedColumnEchelonForm_modular_float()	1008
17.160.2.42 pReducedRowEchelonForm_modular_float()	1008
17.160.2.43 pColumnEchelonForm_modular_int32_t()	1008
17.160.2.44 pRowEchelonForm_modular_int32_t()	1008
17.160.2.45 pReducedColumnEchelonForm_modular_int32_t()	1009
17.160.2.46 pReducedRowEchelonForm_modular_int32_t()	1009
17.160.2.47 Invertin_modular_double()	1009
17.160.2.48 Invert_modular_double()	1009
17.160.2.49 Invert2_modular_double()	1010
17.160.2.50 KrylovElim_modular_double()	1010
17.160.2.51 SpecRankProfile_modular_double()	1010
17.160.2.52 Rank_modular_double()	1010
17.160.2.53 IsSingular_modular_double()	1011
17.160.2.54 Det_modular_double()	1011
17.160.2.55 Solve_modular_double()	1011
17.160.2.56 solveLB_modular_double()	1011
17.160.2.57 solveLB2_modular_double()	1011
17.160.2.58 RandomNullSpaceVector_modular_double()	1012
17.160.2.59 NullSpaceBasis_modular_double()	1012
17.160.2.60 RowRankProfile_modular_double()	1012
17.160.2.61 ColumnRankProfile_modular_double()	1012
17.160.2.62 RankProfileFromLU()	1013
17.160.2.63 LeadingSubmatrixRankProfiles()	1013

17.160.2.64 RowRankProfileSubmatrixIndices_modular_double()	1013
17.160.2.65 ColRankProfileSubmatrixIndices_modular_double()	1013
17.160.2.66 RowRankProfileSubmatrix_modular_double()	1013
17.160.2.67 ColRankProfileSubmatrix_modular_double()	1014
17.160.2.68 getTriangular_modular_double()	1014
17.160.2.69 getTriangularin_modular_double()	1014
17.160.2.70 getEchelonForm_modular_double()	1014
17.160.2.71 getEchelonFormin_modular_double()	1015
17.160.2.72 getEchelonTransform_modular_double()	1015
17.160.2.73 getReducedEchelonForm_modular_double()	1015
17.160.2.74 getReducedEchelonFormin_modular_double()	1016
17.160.2.75 getReducedEchelonTransform_modular_double()	1016
17.160.2.76 PLUQtoEchelonPermutation()	1016
17.161 fpack.dox File Reference	1016
17.162 fpack.h File Reference	1016
17.162.1 Detailed Description	1025
17.162.2 Macro Definition Documentation	1025
17.162.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD	1025
17.162.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD	1025
17.163 fpack.inl File Reference	1026
17.163.1 Macro Definition Documentation	1027
17.163.1.1 __FFLASFFPACK_ffpack_INL	1027
17.164 fpack_bruhatgen.inl File Reference	1027
17.164.1 Macro Definition Documentation	1028
17.164.1.1 __FFLASFFPACK_ffpack_bruhatgen_inl	1028
17.165 fpack_c.h File Reference	1028
17.165.1 Macro Definition Documentation	1031
17.165.1.1 FFPACK_COMPILED	1031
17.165.2 Enumeration Type Documentation	1031
17.165.2.1 FFLAS_C_ORDER	1032
17.165.2.2 FFLAS_C_TRANSPOSE	1032
17.165.2.3 FFLAS_C_UPLO	1032
17.165.2.4 FFLAS_C_DIAG	1032
17.165.2.5 FFLAS_C_SIDE	1032
17.165.2.6 FFPACK_C_LU_TAG	1033
17.165.2.7 FFPACK_C_CHARPOLY_TAG	1033
17.165.2.8 FFPACK_C_MINPOLY_TAG	1033
17.165.3 Function Documentation	1033
17.165.3.1 LAPACKPerm2MathPerm()	1033
17.165.3.2 MathPerm2LAPACKPerm()	1034
17.165.3.3 MatrixApplyS_modular_double()	1034
17.165.3.4 PermApplyS_double()	1034

17.165.3.5 MatrixApplyT_modular_double()	1034
17.165.3.6 PermApplyT_double()	1034
17.165.3.7 composePermutationsLLM()	1035
17.165.3.8 composePermutationsLLL()	1035
17.165.3.9 composePermutationsMLM()	1035
17.165.3.10 cyclic_shift_mathPerm()	1035
17.165.3.11 cyclic_shift_row_modular_double()	1035
17.165.3.12 cyclic_shift_col_modular_double()	1035
17.165.3.13 applyP_modular_double()	1036
17.165.3.14 fgetrsin_modular_double()	1036
17.165.3.15 fgetrs_modular_double()	1036
17.165.3.16 fgesvin_modular_double()	1037
17.165.3.17 fgesv_modular_double()	1037
17.165.3.18 ftrtri_modular_double()	1037
17.165.3.19 trinv_left_modular_double()	1037
17.165.3.20 ftrtrm_modular_double()	1038
17.165.3.21 PLUQ_modular_double()	1038
17.165.3.22 LUdivine_modular_double()	1038
17.165.3.23 LUdivine_small_modular_double()	1038
17.165.3.24 LUdivine_gauss_modular_double()	1039
17.165.3.25 ColumnEchelonForm_modular_double()	1039
17.165.3.26 RowEchelonForm_modular_double()	1039
17.165.3.27 ColumnEchelonForm_modular_float()	1039
17.165.3.28 RowEchelonForm_modular_float()	1040
17.165.3.29 ColumnEchelonForm_modular_int32_t()	1040
17.165.3.30 RowEchelonForm_modular_int32_t()	1040
17.165.3.31 ReducedColumnEchelonForm_modular_double()	1040
17.165.3.32 ReducedRowEchelonForm_modular_double()	1041
17.165.3.33 ReducedColumnEchelonForm_modular_float()	1041
17.165.3.34 ReducedRowEchelonForm_modular_float()	1041
17.165.3.35 ReducedColumnEchelonForm_modular_int32_t()	1041
17.165.3.36 ReducedRowEchelonForm_modular_int32_t()	1042
17.165.3.37 ReducedRowEchelonForm2_modular_double()	1042
17.165.3.38 REF_modular_double()	1042
17.165.3.39 Invertin_modular_double()	1042
17.165.3.40 Invert_modular_double()	1042
17.165.3.41 Invert2_modular_double()	1043
17.165.3.42 KrylovElim_modular_double()	1043
17.165.3.43 SpecRankProfile_modular_double()	1043
17.165.3.44 Rank_modular_double()	1043
17.165.3.45 IsSingular_modular_double()	1044
17.165.3.46 Det_modular_double()	1044

17.165.3.47	Solve_modular_double()	1044
17.165.3.48	solveLB_modular_double()	1044
17.165.3.49	solveLB2_modular_double()	1044
17.165.3.50	RandomNullSpaceVector_modular_double()	1045
17.165.3.51	NullSpaceBasis_modular_double()	1045
17.165.3.52	RowRankProfile_modular_double()	1045
17.165.3.53	ColumnRankProfile_modular_double()	1045
17.165.3.54	RankProfileFromLU()	1046
17.165.3.55	LeadingSubmatrixRankProfiles()	1046
17.165.3.56	RowRankProfileSubmatrixIndices_modular_double()	1046
17.165.3.57	ColRankProfileSubmatrixIndices_modular_double()	1046
17.165.3.58	RowRankProfileSubmatrix_modular_double()	1047
17.165.3.59	ColRankProfileSubmatrix_modular_double()	1047
17.165.3.60	getTriangular_modular_double()	1047
17.165.3.61	getTriangularin_modular_double()	1047
17.165.3.62	getEchelonForm_modular_double()	1047
17.165.3.63	getEchelonFormin_modular_double()	1048
17.165.3.64	getEchelonTransform_modular_double()	1048
17.165.3.65	getReducedEchelonForm_modular_double()	1048
17.165.3.66	getReducedEchelonFormin_modular_double()	1049
17.165.3.67	getReducedEchelonTransform_modular_double()	1049
17.165.3.68	PLUQtoEchelonPermutation()	1049
17.166	ffpack_charpoly.inl File Reference	1049
17.166.1	Macro Definition Documentation	1050
17.166.1.1	__FFLASFFPACK_charpoly_INL	1050
17.167	ffpack_charpoly_danilevski.inl File Reference	1050
17.167.1	Macro Definition Documentation	1050
17.167.1.1	__FFLASFFPACK_ffpack_charpoly_danilveski_INL	1051
17.168	ffpack_charpoly_kgfast.inl File Reference	1051
17.168.1	Macro Definition Documentation	1051
17.168.1.1	__FFLASFFPACK_ffpack_charpoly_kgfast_INL	1051
17.169	ffpack_charpoly_kgfastgeneralized.inl File Reference	1051
17.169.1	Macro Definition Documentation	1052
17.169.1.1	__FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL	1052
17.170	ffpack_charpoly_kglu.inl File Reference	1052
17.170.1	Macro Definition Documentation	1052
17.170.1.1	__FFLASFFPACK_ffpack_charpoly_kglu_INL	1052
17.171	ffpack_charpoly_mp.inl File Reference	1052
17.171.1	Macro Definition Documentation	1053
17.171.1.1	__FFPACK_charpoly_mp_INL	1053
17.172	ffpack_det_mp.inl File Reference	1053
17.172.1	Macro Definition Documentation	1053

17.172.1.1	<a href="#">__FFPACK_det_mp_INL</a>	1053
17.173	<a href="#">ffpack_echelonforms.inl</a> File Reference	1054
17.173.1	Macro Definition Documentation	1055
17.173.1.1	<a href="#">__FFLASFFPACK_ffpack_echelon_forms_INL</a>	1055
17.173.1.2	<a href="#">__FFLASFFPACK_GAUSSJORDAN_BASECASE</a>	1055
17.174	<a href="#">ffpack_fgesv.inl</a> File Reference	1055
17.174.1	Macro Definition Documentation	1055
17.174.1.1	<a href="#">__FFLASFFPACK_ffpack_fgesv_INL</a>	1055
17.175	<a href="#">ffpack_fgetrs.inl</a> File Reference	1056
17.175.1	Macro Definition Documentation	1056
17.175.1.1	<a href="#">__FFLASFFPACK_ffpack_fgetrs_INL</a>	1056
17.176	<a href="#">ffpack_frobenius.inl</a> File Reference	1056
17.177	<a href="#">ffpack_fsytrf.inl</a> File Reference	1057
17.177.1	Macro Definition Documentation	1058
17.177.1.1	<a href="#">__FFLASFFPACK_ffpack_fsytrf_INL</a>	1058
17.178	<a href="#">ffpack_frssyr2k.inl</a> File Reference	1058
17.178.1	Macro Definition Documentation	1058
17.178.1.1	<a href="#">__FFLASFFPACK_ffpack_frssyr2k_INL</a>	1059
17.179	<a href="#">ffpack_ftrstr.inl</a> File Reference	1059
17.179.1	Macro Definition Documentation	1059
17.179.1.1	<a href="#">__FFLASFFPACK_ffpack_ftrstr_INL</a>	1059
17.180	<a href="#">ffpack_ftrtr.inl</a> File Reference	1059
17.180.1	Macro Definition Documentation	1060
17.180.1.1	<a href="#">ENABLE_ALL_CHECKINGS</a>	1060
17.180.1.2	<a href="#">__FFLASFFPACK_ffpack_ftrtr_INL</a>	1060
17.181	<a href="#">ffpack_inst.C</a> File Reference	1060
17.181.1	Macro Definition Documentation	1060
17.181.1.1	<a href="#">__FFPACK_INST_C</a>	1060
17.181.1.2	<a href="#">FFLAS_COMPILED</a>	1060
17.181.1.3	<a href="#">INST_OR_DECL</a>	1061
17.181.1.4	<a href="#">FFLAS_FIELD</a> [1/2]	1061
17.181.1.5	<a href="#">FFLAS_ELT</a> [1/6]	1061
17.181.1.6	<a href="#">FFLAS_ELT</a> [2/6]	1061
17.181.1.7	<a href="#">FFLAS_ELT</a> [3/6]	1061
17.181.1.8	<a href="#">FFLAS_FIELD</a> [2/2]	1061
17.181.1.9	<a href="#">FFLAS_ELT</a> [4/6]	1061
17.181.1.10	<a href="#">FFLAS_ELT</a> [5/6]	1061
17.181.1.11	<a href="#">FFLAS_ELT</a> [6/6]	1061
17.182	<a href="#">ffpack_inst.h</a> File Reference	1061
17.182.1	Macro Definition Documentation	1062
17.182.1.1	<a href="#">FFLAS_COMPILED</a>	1062
17.182.1.2	<a href="#">INST_OR_DECL</a>	1062

17.182.1.3 FFLAS_FIELD [1/2]	1062
17.182.1.4 FFLAS_ELT [1/6]	1062
17.182.1.5 FFLAS_ELT [2/6]	1062
17.182.1.6 FFLAS_ELT [3/6]	1062
17.182.1.7 FFLAS_FIELD [2/2]	1062
17.182.1.8 FFLAS_ELT [4/6]	1062
17.182.1.9 FFLAS_ELT [5/6]	1062
17.182.1.10 FFLAS_ELT [6/6]	1062
17.183 fpack_inst_implem.inl File Reference	1063
17.184 fpack_invert.inl File Reference	1066
17.184.1 Macro Definition Documentation	1066
17.184.1.1 __FFLASFFPACK_fpack_invert_INL	1066
17.185 fpack_krylovelim.inl File Reference	1066
17.185.1 Macro Definition Documentation	1066
17.185.1.1 __FFLASFFPACK_fpack_krylovelim_INL	1067
17.186 fpack_ludivine.inl File Reference	1067
17.186.1 Macro Definition Documentation	1067
17.186.1.1 __FFLASFFPACK_fpack_ludivine_INL	1067
17.187 fpack_ludivine_mp.inl File Reference	1068
17.187.1 Macro Definition Documentation	1068
17.187.1.1 __FFPACK_ludivine_mp_INL	1068
17.188 fpack_minpoly.inl File Reference	1068
17.188.1 Macro Definition Documentation	1069
17.188.1.1 __FFLASFFPACK_fpack_minpoly_INL	1069
17.189 fpack_permutation.inl File Reference	1069
17.189.1 Macro Definition Documentation	1071
17.189.1.1 __FFLASFFPACK_fpack_permutation_INL	1071
17.189.1.2 FFLASFFPACK_PERM_BKSIZE	1071
17.190 fpack_pluq.inl File Reference	1071
17.190.1 Macro Definition Documentation	1072
17.190.1.1 __FFLASFFPACK_fpack_pluq_INL	1072
17.190.1.2 CROUT	1072
17.191 fpack_pluq_mp.inl File Reference	1072
17.191.1 Macro Definition Documentation	1073
17.191.1.1 __FFPACK_pluq_mp_INL	1073
17.192 fpack_ppluq.inl File Reference	1073
17.192.1 Macro Definition Documentation	1073
17.192.1.1 __FFLASFFPACK_fpack_ppluq_INL	1073
17.192.1.2 __FFLAS__TRSM_READONLY	1073
17.192.1.3 PBASECASE_K	1073
17.193 fpack_rankprofiles.inl File Reference	1074
17.193.1 Macro Definition Documentation	1075

17.193.1.1 <a href="#">__FFLASFFPACK_ffpack_rank_profiles_INL</a>	1075
17.194 <a href="#">fgemm_classical.inl</a> File Reference	1075
17.194.1 Macro Definition Documentation	1075
17.194.1.1 <a href="#">__FFLASFFPACK_fflas_fflas_fgemm_classical_INL</a>	1075
17.195 <a href="#">fgemm_classical_mp.inl</a> File Reference	1075
17.195.1 Detailed Description	1077
17.195.2 Macro Definition Documentation	1077
17.195.2.1 <a href="#">__FFPACK_fgemm_classical_INL</a>	1077
17.196 <a href="#">fgemm_winograd.inl</a> File Reference	1077
17.196.1 Macro Definition Documentation	1078
17.196.1.1 <a href="#">__FFLASFFPACK_fflas_fflas_fgemm_winograd_INL</a>	1078
17.196.1.2 NEWWINO	1078
17.197 <a href="#">field-traits.h</a> File Reference	1079
17.197.1 Detailed Description	1081
17.198 <a href="#">field.doxy</a> File Reference	1081
17.199 <a href="#">flimits.h</a> File Reference	1081
17.199.1 Function Documentation	1082
17.199.1.1 <a href="#">in_range()</a> [1/3]	1082
17.199.1.2 <a href="#">in_range()</a> [2/3]	1082
17.199.1.3 <a href="#">in_range()</a> [3/3]	1082
17.200 <a href="#">fsyrk.C</a> File Reference	1082
17.200.1 Macro Definition Documentation	1082
17.200.1.1 CUBE	1083
17.200.1.2 GFOPS	1083
17.200.2 Typedef Documentation	1083
17.200.2.1 TTimer	1083
17.200.3 Function Documentation	1083
17.200.3.1 <a href="#">main()</a>	1083
17.201 <a href="#">fsytrf.C</a> File Reference	1083
17.201.1 Macro Definition Documentation	1083
17.201.1.1 CUBE	1084
17.201.1.2 GFOPS	1084
17.201.2 Typedef Documentation	1084
17.201.2.1 TTimer	1084
17.201.3 Function Documentation	1084
17.201.3.1 <a href="#">main()</a>	1084
17.202 <a href="#">ftrtri.C</a> File Reference	1084
17.202.1 Macro Definition Documentation	1084
17.202.1.1 CUBE	1085
17.202.1.2 GFOPS	1085
17.202.2 Typedef Documentation	1085
17.202.2.1 TTimer	1085

17.202.3 Function Documentation	1085
17.202.3.1 main()	1085
17.203 hyb_zo.h File Reference	1085
17.204 hyb_zo_pspmm.inl File Reference	1085
17.204.1 Macro Definition Documentation	1086
17.204.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL	1086
17.205 hyb_zo_pspmv.inl File Reference	1086
17.205.1 Macro Definition Documentation	1086
17.205.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL	1086
17.206 hyb_zo_spm্ম.inl File Reference	1086
17.206.1 Macro Definition Documentation	1087
17.206.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spm্ম_INL	1087
17.207 hyb_zo_spmmv.inl File Reference	1087
17.207.1 Macro Definition Documentation	1087
17.207.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmmv_INL	1087
17.208 hyb_zo_utils.inl File Reference	1088
17.208.1 Macro Definition Documentation	1088
17.208.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL	1088
17.209 igemm.doxy File Reference	1088
17.210 igemm.h File Reference	1088
17.211 igemm.inl File Reference	1089
17.211.1 Macro Definition Documentation	1089
17.211.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL	1089
17.212 igemm_kernels.h File Reference	1089
17.213 igemm_kernels.inl File Reference	1090
17.213.1 Macro Definition Documentation	1091
17.213.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL	1091
17.214 igemm_tools.h File Reference	1091
17.215 igemm_tools.inl File Reference	1091
17.215.1 Macro Definition Documentation	1091
17.215.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL	1091
17.216 interfaces.doxy File Reference	1092
17.217 kaapi_routines.inl File Reference	1092
17.217.1 Macro Definition Documentation	1092
17.217.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL	1092
17.218 lapack.C File Reference	1092
17.218.1 Macro Definition Documentation	1092
17.218.1.1 __FFLASFFPACK_CONFIGURATION	1092
17.218.1.2 __FFLASFFPACK_HAVE_LAPACK	1092
17.218.2 Function Documentation	1092
17.218.2.1 main()	1092
17.219 mainpage.doxy File Reference	1093

17.220 Matio.h File Reference . . . . .	1093
17.220.1 Function Documentation . . . . .	1093
17.220.1.1 read_field() . . . . .	1093
17.220.1.2 write_field() . . . . .	1093
17.221 matmul.C File Reference . . . . .	1093
17.221.1 Function Documentation . . . . .	1093
17.221.1.1 main() . . . . .	1094
17.222 matmul.doxy File Reference . . . . .	1094
17.223 parallel.h File Reference . . . . .	1094
17.223.1 Macro Definition Documentation . . . . .	1095
17.223.1.1 __FFLASFFPACK_SEQUENTIAL . . . . .	1095
17.223.1.2 index_t . . . . .	1095
17.223.1.3 TASK . . . . .	1095
17.223.1.4 WAIT . . . . .	1095
17.223.1.5 CHECK_DEPENDENCIES . . . . .	1095
17.223.1.6 BARRIER . . . . .	1095
17.223.1.7 PAR_BLOCK . . . . .	1095
17.223.1.8 SYNCH_GROUP . . . . .	1095
17.223.1.9 THREAD_INDEX . . . . .	1095
17.223.1.10 NUM_THREADS . . . . .	1096
17.223.1.11 SET_THREADS . . . . .	1096
17.223.1.12 MAX_THREADS . . . . .	1096
17.223.1.13 READ . . . . .	1096
17.223.1.14 WRITE . . . . .	1096
17.223.1.15 READWRITE . . . . .	1096
17.223.1.16 CONSTREFERENCE . . . . .	1096
17.223.1.17 VALUE . . . . .	1096
17.223.1.18 BEGIN_PARALLEL_MAIN . . . . .	1096
17.223.1.19 END_PARALLEL_MAIN . . . . .	1096
17.223.1.20 FORBLOCK1D . . . . .	1097
17.223.1.21 FOR1D . . . . .	1097
17.223.1.22 PARFORBLOCK1D . . . . .	1097
17.223.1.23 PARFOR1D . . . . .	1097
17.223.1.24 FORBLOCK2D . . . . .	1097
17.223.1.25 FOR2D . . . . .	1098
17.223.1.26 PARFORBLOCK2D . . . . .	1098
17.223.1.27 PARFOR2D . . . . .	1098
17.223.1.28 COMMA . . . . .	1098
17.223.1.29 MODE . . . . .	1098
17.223.1.30 RETURNPARAM . . . . .	1098
17.223.1.31 NUMARGS . . . . .	1098
17.223.1.32 PP_NARG_ . . . . .	1099

17.223.1.33 PP_ARG_N . . . . .	1099
17.223.1.34 PP_RSEQ_N . . . . .	1100
17.223.1.35 NOSPLIT . . . . .	1100
17.223.1.36 splitting_0 . . . . .	1100
17.223.1.37 splitting_1 . . . . .	1100
17.223.1.38 splitting_2 . . . . .	1100
17.223.1.39 splitting_3 . . . . .	1100
17.223.1.40 splitt . . . . .	1101
17.223.1.41 SPLITTER . . . . .	1101
17.224 pfgemm_variants.inl File Reference . . . . .	1101
17.225 pfgemv.inl File Reference . . . . .	1102
17.226 pluq.C File Reference . . . . .	1102
17.226.1 Macro Definition Documentation . . . . .	1102
17.226.1.1 CUBE . . . . .	1103
17.226.1.2 GFOPS . . . . .	1103
17.226.2 Typedef Documentation . . . . .	1103
17.226.2.1 TTimer . . . . .	1103
17.226.3 Function Documentation . . . . .	1103
17.226.3.1 main() . . . . .	1103
17.227 pluq.C File Reference . . . . .	1103
17.227.1 Function Documentation . . . . .	1103
17.227.1.1 main() . . . . .	1103
17.228 rank.C File Reference . . . . .	1104
17.228.1 Function Documentation . . . . .	1104
17.228.1.1 main() . . . . .	1104
17.229 read_sparse.h File Reference . . . . .	1104
17.229.1 Macro Definition Documentation . . . . .	1105
17.229.1.1 DNS_BIN_VER . . . . .	1105
17.229.1.2 mask_t . . . . .	1105
17.230 regression-check.C File Reference . . . . .	1105
17.230.1 Function Documentation . . . . .	1105
17.230.1.1 check1() . . . . .	1105
17.230.1.2 check2() . . . . .	1106
17.230.1.3 check3() . . . . .	1106
17.230.1.4 check4() . . . . .	1106
17.230.1.5 checkZeroDimCharpoly() . . . . .	1106
17.230.1.6 checkZeroDimMinPoly() . . . . .	1106
17.230.1.7 gf2ModularBalanced() . . . . .	1106
17.230.1.8 main() . . . . .	1106
17.231 rns-double-elt.h File Reference . . . . .	1106
17.231.1 Detailed Description . . . . .	1107
17.232 rns-double-recint.inl File Reference . . . . .	1107

17.232.1 Macro Definition Documentation	1107
17.232.1.1 __FFLASFFPACK_field_rns_double_recint_INL	1107
17.233 rns-double.h File Reference	1107
17.233.1 Detailed Description	1108
17.233.2 Macro Definition Documentation	1108
17.233.2.1 ROUND_DOWN	1108
17.234 rns-double.inl File Reference	1108
17.234.1 Macro Definition Documentation	1108
17.234.1.1 __FFLASFFPACK_field_rns_double_INL	1108
17.235 rns-integer-mod.h File Reference	1108
17.235.1 Detailed Description	1109
17.236 rns-integer.h File Reference	1109
17.236.1 Detailed Description	1110
17.237 rns.h File Reference	1110
17.238 rns.inl File Reference	1110
17.238.1 Macro Definition Documentation	1110
17.238.1.1 __FFLASFFPACK_field_rns_INL	1110
17.239 schedule_bini.inl File Reference	1110
17.239.1 Detailed Description	1111
17.239.2 Macro Definition Documentation	1111
17.239.2.1 __FFLASFFPACK_fgemm_bini_INL	1111
17.240 schedule_winograd.inl File Reference	1111
17.240.1 Macro Definition Documentation	1111
17.240.1.1 __FFLASFFPACK_fgemm_winograd_INL	1111
17.241 schedule_winograd_acc.inl File Reference	1112
17.241.1 Macro Definition Documentation	1112
17.241.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL	1112
17.242 schedule_winograd_acc_ip.inl File Reference	1112
17.242.1 Macro Definition Documentation	1113
17.242.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL	1113
17.243 schedule_winograd_ip.inl File Reference	1113
17.243.1 Macro Definition Documentation	1114
17.243.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL	1114
17.244 sell.h File Reference	1114
17.245 sell_pspmv.inl File Reference	1114
17.245.1 Macro Definition Documentation	1115
17.245.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL	1115
17.246 sell_spmv.inl File Reference	1115
17.246.1 Macro Definition Documentation	1116
17.246.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL	1116
17.247 sell_utils.inl File Reference	1116
17.247.1 Macro Definition Documentation	1116

17.247.1.1 <a href="#">__FFLASFFPACK_fflas_sparse_sell_utils_INL</a> . . . . .	1116
17.248 <a href="#">simd.doxy</a> File Reference . . . . .	1116
17.249 <a href="#">simd128.inl</a> File Reference . . . . .	1116
17.249.1 Macro Definition Documentation . . . . .	1117
17.249.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_INL</a> . . . . .	1117
17.249.2 Typedef Documentation . . . . .	1117
17.249.2.1 <a href="#">Simd128</a> . . . . .	1117
17.250 <a href="#">simd128_double.inl</a> File Reference . . . . .	1117
17.250.1 Macro Definition Documentation . . . . .	1117
17.250.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL</a> . . . . .	1117
17.251 <a href="#">simd128_float.inl</a> File Reference . . . . .	1118
17.251.1 Macro Definition Documentation . . . . .	1118
17.251.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL</a> . . . . .	1118
17.252 <a href="#">simd128_int16.inl</a> File Reference . . . . .	1118
17.252.1 Macro Definition Documentation . . . . .	1118
17.252.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL</a> . . . . .	1118
17.253 <a href="#">simd128_int32.inl</a> File Reference . . . . .	1118
17.253.1 Macro Definition Documentation . . . . .	1119
17.253.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL</a> . . . . .	1119
17.254 <a href="#">simd128_int64.inl</a> File Reference . . . . .	1119
17.254.1 Macro Definition Documentation . . . . .	1119
17.254.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL</a> . . . . .	1119
17.254.1.2 <a href="#">vect_t</a> . . . . .	1119
17.255 <a href="#">simd256.inl</a> File Reference . . . . .	1119
17.255.1 Macro Definition Documentation . . . . .	1120
17.255.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_INL</a> . . . . .	1120
17.255.2 Typedef Documentation . . . . .	1120
17.255.2.1 <a href="#">Simd256</a> . . . . .	1120
17.256 <a href="#">simd256_double.inl</a> File Reference . . . . .	1120
17.256.1 Macro Definition Documentation . . . . .	1120
17.256.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL</a> . . . . .	1120
17.257 <a href="#">simd256_float.inl</a> File Reference . . . . .	1121
17.257.1 Macro Definition Documentation . . . . .	1121
17.257.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL</a> . . . . .	1121
17.258 <a href="#">simd256_int16.inl</a> File Reference . . . . .	1121
17.258.1 Macro Definition Documentation . . . . .	1121
17.258.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL</a> . . . . .	1121
17.259 <a href="#">simd256_int32.inl</a> File Reference . . . . .	1121
17.259.1 Macro Definition Documentation . . . . .	1122
17.259.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL</a> . . . . .	1122
17.260 <a href="#">simd256_int64.inl</a> File Reference . . . . .	1122
17.260.1 Macro Definition Documentation . . . . .	1122

17.260.1.1	<a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL</a>	1122
17.260.1.2	<a href="#">vect_t</a>	1122
17.261	<a href="#">simd512.inl File Reference</a>	1123
17.261.1	<a href="#">Macro Definition Documentation</a>	1123
17.261.1.1	<a href="#">__FFLASFFPACK_simd512_INL</a>	1123
17.261.2	<a href="#">Typedef Documentation</a>	1123
17.261.2.1	<a href="#">Simd512</a>	1123
17.262	<a href="#">simd512_double.inl File Reference</a>	1123
17.262.1	<a href="#">Macro Definition Documentation</a>	1123
17.262.1.1	<a href="#">__FFLASFFPACK_simd512_double_INL</a>	1124
17.263	<a href="#">simd512_float.inl File Reference</a>	1124
17.263.1	<a href="#">Macro Definition Documentation</a>	1124
17.263.1.1	<a href="#">__FFLASFFPACK_simd512_float_INL</a>	1124
17.264	<a href="#">simd512_int32.inl File Reference</a>	1124
17.264.1	<a href="#">Macro Definition Documentation</a>	1124
17.264.1.1	<a href="#">__FFLASFFPACK_simd512_int32_INL</a>	1124
17.265	<a href="#">simd512_int64.inl File Reference</a>	1125
17.265.1	<a href="#">Macro Definition Documentation</a>	1125
17.265.1.1	<a href="#">_simd512_int64_INL</a>	1125
17.265.1.2	<a href="#">vect_t</a>	1125
17.266	<a href="#">simd_modular.inl File Reference</a>	1125
17.267	<a href="#">solve.C File Reference</a>	1125
17.267.1	<a href="#">Function Documentation</a>	1125
17.267.1.1	<a href="#">main()</a>	1126
17.268	<a href="#">sparse_matrix_traits.h File Reference</a>	1126
17.269	<a href="#">test-charpoly-check.C File Reference</a>	1127
17.269.1	<a href="#">Macro Definition Documentation</a>	1127
17.269.1.1	<a href="#">ENABLE_CHECKER_charpoly</a>	1128
17.269.1.2	<a href="#">TIME_CHECKER_CHARPOLY</a>	1128
17.269.2	<a href="#">Function Documentation</a>	1128
17.269.2.1	<a href="#">printPolynomial()</a>	1128
17.269.2.2	<a href="#">main()</a>	1128
17.270	<a href="#">test-charpoly.C File Reference</a>	1128
17.270.1	<a href="#">Function Documentation</a>	1128
17.270.1.1	<a href="#">launch_test()</a>	1128
17.270.1.2	<a href="#">run_with_field()</a>	1129
17.270.1.3	<a href="#">main()</a>	1129
17.271	<a href="#">test-compressQ.C File Reference</a>	1129
17.271.1	<a href="#">Typedef Documentation</a>	1129
17.271.1.1	<a href="#">Field</a>	1129
17.271.2	<a href="#">Function Documentation</a>	1130
17.271.2.1	<a href="#">printvect()</a>	1130

17.271.2.2 main()	1130
17.272 test-det-check.C File Reference	1130
17.272.1 Macro Definition Documentation	1130
17.272.1.1 ENABLE_CHECKER_Det	1130
17.272.1.2 TIME_CHECKER_Det	1130
17.272.2 Function Documentation	1130
17.272.2.1 main()	1131
17.273 test-det.C File Reference	1131
17.273.1 Function Documentation	1131
17.273.1.1 test_det()	1131
17.273.1.2 main()	1131
17.274 test-echelon.C File Reference	1131
17.274.1 Macro Definition Documentation	1132
17.274.1.1 __FFLASFFPACK_SEQUENTIAL	1132
17.274.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE	1132
17.274.1.3 __FFLASFFPACK_PLUQ_THRESHOLD	1132
17.274.2 Function Documentation	1132
17.274.2.1 test_colechelon()	1132
17.274.2.2 test_rowechelon()	1133
17.274.2.3 test_redcolechelon()	1133
17.274.2.4 test_redrowechelon()	1133
17.274.2.5 run_with_field()	1133
17.274.2.6 main()	1134
17.275 test-fadd.C File Reference	1134
17.275.1 Function Documentation	1134
17.275.1.1 test_fadd()	1134
17.275.1.2 test_faddin()	1134
17.275.1.3 test_fsub()	1135
17.275.1.4 test_fsubin()	1135
17.275.1.5 main()	1135
17.276 test-fdot.C File Reference	1135
17.276.1 Macro Definition Documentation	1136
17.276.1.1 ENABLE_ALL_CHECKINGS	1136
17.276.2 Function Documentation	1136
17.276.2.1 check_fdot()	1136
17.276.2.2 run_with_field()	1136
17.276.2.3 run_with_Integer()	1136
17.276.2.4 main()	1136
17.277 test-fgemm-check.C File Reference	1136
17.277.1 Macro Definition Documentation	1137
17.277.1.1 ENABLE_ALL_CHECKINGS	1137
17.277.2 Function Documentation	1137

17.277.2.1 launch_MM_dispatch()	1137
17.277.2.2 run_with_field()	1137
17.277.2.3 main()	1138
17.278 test-fgemm.C File Reference	1138
17.278.1 Macro Definition Documentation	1138
17.278.1.1 ENABLE_CHECKER_fgemm	1138
17.278.2 Function Documentation	1138
17.278.2.1 check_MM()	1139
17.278.2.2 launch_MM()	1139
17.278.2.3 launch_MM_dispatch()	1139
17.278.2.4 run_with_field()	1140
17.278.2.5 main()	1140
17.279 test-fgemv.C File Reference	1140
17.279.1 Function Documentation	1140
17.279.1.1 check_MV()	1141
17.279.1.2 launch_MV()	1141
17.279.1.3 launch_MV_dispatch()	1141
17.279.1.4 run_with_field()	1141
17.279.1.5 main()	1142
17.280 test-fger.C File Reference	1142
17.280.1 Macro Definition Documentation	1142
17.280.1.1 TIME	1142
17.280.2 Function Documentation	1142
17.280.2.1 check_fger()	1142
17.280.2.2 launch_fger()	1143
17.280.2.3 launch_fger_dispatch()	1143
17.280.2.4 run_with_field()	1143
17.280.2.5 main()	1143
17.281 test-fgesv.C File Reference	1144
17.281.1 Function Documentation	1144
17.281.1.1 test_square_fgesv()	1144
17.281.1.2 test_rect_fgesv()	1144
17.281.1.3 run_with_field()	1144
17.281.1.4 main()	1145
17.282 test-finit.C File Reference	1145
17.282.1 Function Documentation	1145
17.282.1.1 test_freduce()	1145
17.282.1.2 run_with_field()	1146
17.282.1.3 main()	1146
17.283 test-fscal.C File Reference	1146
17.283.1 Function Documentation	1146
17.283.1.1 test_fscal() [1/2]	1146

17.283.1.2 test_fscal() [2/2]	1147
17.283.1.3 test_fscaln() [1/2]	1147
17.283.1.4 test_fscaln() [2/2]	1147
17.283.1.5 main()	1147
17.284 test-fsyr2k.C File Reference	1147
17.284.1 Macro Definition Documentation	1148
17.284.1.1 ENABLE_ALL_CHECKINGS	1148
17.284.2 Function Documentation	1148
17.284.2.1 check_fsyr2k()	1148
17.284.2.2 run_with_field()	1148
17.284.2.3 main()	1148
17.285 test-fsyrk.C File Reference	1149
17.285.1 Macro Definition Documentation	1149
17.285.1.1 ENABLE_ALL_CHECKINGS	1149
17.285.2 Function Documentation	1149
17.285.2.1 check_fsyrk()	1149
17.285.2.2 check_fsyrk_diag()	1150
17.285.2.3 check_fsyrk_bkdiag()	1150
17.285.2.4 check_computeS1S2()	1150
17.285.2.5 run_with_field()	1150
17.285.2.6 main()	1150
17.286 test-fsytrf.C File Reference	1151
17.286.1 Function Documentation	1151
17.286.1.1 operator<<()	1151
17.286.1.2 test_RPM_fsytrf()	1151
17.286.1.3 test_generic_fsytrf()	1152
17.286.1.4 run_with_field()	1152
17.286.1.5 main()	1152
17.287 test-frm.C File Reference	1152
17.287.1 Macro Definition Documentation	1153
17.287.1.1 __FFLASFFPACK_SEQUENTIAL	1153
17.287.2 Function Documentation	1153
17.287.2.1 check_frm()	1153
17.287.2.2 run_with_field()	1153
17.287.2.3 main()	1153
17.288 test-frm.C File Reference	1153
17.288.1 Macro Definition Documentation	1154
17.288.1.1 __FFLASFFPACK_SEQUENTIAL	1154
17.288.1.2 ENABLE_ALL_CHECKINGS	1154
17.288.2 Function Documentation	1154
17.288.2.1 check_frmv()	1154
17.288.2.2 run_with_field()	1154

17.288.2.3 main()	1154
17.289 test-ftsm-check.C File Reference	1155
17.289.1 Macro Definition Documentation	1155
17.289.1.1 ENABLE_ALL_CHECKINGS	1155
17.289.2 Function Documentation	1155
17.289.2.1 main()	1155
17.290 test-ftsm.C File Reference	1155
17.290.1 Macro Definition Documentation	1156
17.290.1.1 __FFLASFFPACK_SEQUENTIAL	1156
17.290.1.2 ENABLE_ALL_CHECKINGS	1156
17.290.2 Function Documentation	1156
17.290.2.1 check_ftsm()	1156
17.290.2.2 run_with_field()	1156
17.290.2.3 main()	1156
17.291 test-ftssyr2k.C File Reference	1157
17.291.1 Macro Definition Documentation	1157
17.291.1.1 ENABLE_ALL_CHECKINGS	1157
17.291.2 Function Documentation	1157
17.291.2.1 check_ftssyr2k()	1157
17.291.2.2 run_with_field()	1157
17.291.2.3 main()	1158
17.292 test-ftsstr.C File Reference	1158
17.292.1 Macro Definition Documentation	1158
17.292.1.1 ENABLE_ALL_CHECKINGS	1158
17.292.2 Function Documentation	1158
17.292.2.1 check_ftsstr()	1158
17.292.2.2 run_with_field()	1159
17.292.2.3 main()	1159
17.293 test-ftsv.C File Reference	1159
17.293.1 Macro Definition Documentation	1159
17.293.1.1 __FFLASFFPACK_SEQUENTIAL	1159
17.293.1.2 ENABLE_ALL_CHECKINGS	1160
17.293.2 Function Documentation	1160
17.293.2.1 check_ftsv()	1160
17.293.2.2 run_with_field()	1160
17.293.2.3 main()	1160
17.294 test-ftsrti.C File Reference	1160
17.294.1 Macro Definition Documentation	1161
17.294.1.1 __FFLASFFPACK_SEQUENTIAL	1161
17.294.1.2 ENABLE_ALL_CHECKINGS	1161
17.294.2 Function Documentation	1161
17.294.2.1 check_ftsrti()	1161

17.294.2.2 run_with_field()	1161
17.294.2.3 main()	1161
17.295 test-interfaces-c.c File Reference	1161
17.295.1 Function Documentation	1162
17.295.1.1 main()	1162
17.296 test-invert-check.C File Reference	1162
17.296.1 Macro Definition Documentation	1162
17.296.1.1 ENABLE_ALL_CHECKINGS	1162
17.296.2 Function Documentation	1162
17.296.2.1 main()	1162
17.297 test-io.C File Reference	1162
17.297.1 Function Documentation	1163
17.297.1.1 run_with_field()	1163
17.297.1.2 main()	1163
17.298 test-lu.C File Reference	1163
17.298.1 Macro Definition Documentation	1164
17.298.1.1 BASECASE_K	1164
17.298.1.2 __FFLASFFPACK_SEQUENTIAL	1164
17.298.1.3 __LUDIVINE_CUTOFF	1164
17.298.2 Function Documentation	1164
17.298.2.1 test_LUdivine()	1165
17.298.2.2 verifPLUQ()	1165
17.298.2.3 test_pluq()	1166
17.298.2.4 launch_test()	1166
17.298.2.5 run_with_field()	1167
17.298.2.6 main()	1167
17.298.3 Variable Documentation	1167
17.298.3.1 tperm	1167
17.298.3.2 tgemm	1167
17.298.3.3 tBC	1167
17.298.3.4 ttrsm	1167
17.298.3.5 trest	1167
17.298.3.6 timtot	1167
17.298.3.7 mvcnt	1167
17.299 test-maxdelayeddim.C File Reference	1168
17.299.1 Macro Definition Documentation	1168
17.299.1.1 MAX_WITH_SIZE_T	1168
17.299.2 Function Documentation	1168
17.299.2.1 test()	1168
17.299.2.2 main()	1168
17.300 test-minpoly.C File Reference	1168
17.300.1 Function Documentation	1169

17.300.1.1	<a href="#">check_minpoly()</a>	1169
17.300.1.2	<a href="#">run_with_field()</a>	1169
17.300.1.3	<a href="#">main()</a>	1169
17.301	<a href="#">test-multifile1.C File Reference</a>	1169
17.302	<a href="#">test-multifile2.C File Reference</a>	1169
17.302.1	<a href="#">Function Documentation</a>	1170
17.302.1.1	<a href="#">main()</a>	1170
17.303	<a href="#">test-nullspace.C File Reference</a>	1170
17.303.1	<a href="#">Function Documentation</a>	1170
17.303.1.1	<a href="#">checkingMessage()</a>	1170
17.303.1.2	<a href="#">readOrRandomMatrixWithRankAndRandomRPM()</a>	1170
17.303.1.3	<a href="#">test_nullspace()</a>	1171
17.303.1.4	<a href="#">run_with_field()</a>	1171
17.303.1.5	<a href="#">main()</a>	1171
17.304	<a href="#">test-permutations.C File Reference</a>	1171
17.304.1	<a href="#">Function Documentation</a>	1171
17.304.1.1	<a href="#">checkMonotonicApplyP()</a>	1172
17.304.1.2	<a href="#">main()</a>	1172
17.304.2	<a href="#">Variable Documentation</a>	1172
17.304.2.1	<a href="#">tperm</a>	1172
17.304.2.2	<a href="#">tgemm</a>	1172
17.304.2.3	<a href="#">tBC</a>	1172
17.304.2.4	<a href="#">ttrsm</a>	1172
17.304.2.5	<a href="#">trest</a>	1172
17.304.2.6	<a href="#">timtot</a>	1172
17.305	<a href="#">test-pluq-check.C File Reference</a>	1172
17.305.1	<a href="#">Macro Definition Documentation</a>	1173
17.305.1.1	<a href="#">ENABLE_ALL_CHECKINGS</a>	1173
17.305.2	<a href="#">Function Documentation</a>	1173
17.305.2.1	<a href="#">main()</a>	1173
17.306	<a href="#">test-quasisep.C File Reference</a>	1173
17.306.1	<a href="#">Function Documentation</a>	1173
17.306.1.1	<a href="#">test_BruhatGenerator()</a>	1174
17.306.1.2	<a href="#">launch_test()</a>	1174
17.306.1.3	<a href="#">testLTQSRPM()</a>	1174
17.306.1.4	<a href="#">run_with_field()</a>	1174
17.306.1.5	<a href="#">main()</a>	1174
17.307	<a href="#">test-rankprofiles.C File Reference</a>	1174
17.307.1	<a href="#">Macro Definition Documentation</a>	1175
17.307.1.1	<a href="#">__FFLASFFPACK_SEQUENTIAL</a>	1175
17.307.2	<a href="#">Function Documentation</a>	1175
17.307.2.1	<a href="#">run_with_field()</a>	1175

17.307.2.2 main()	1175
17.308 test-rpm.C File Reference	1175
17.308.1 Function Documentation	1176
17.308.1.1 checkRPM()	1176
17.308.1.2 checkSymmetricRPM()	1176
17.308.1.3 main()	1176
17.309 test-simd.C File Reference	1176
17.309.1 Macro Definition Documentation	1177
17.309.1.1 _TEST_ONE	1178
17.309.1.2 TEST_ONE_OP	1178
17.309.1.3 TEST_ONE_OP_WZ	1178
17.309.1.4 TEST_IMPL	1178
17.309.2 Function Documentation	1178
17.309.2.1 check_eq() [1/2]	1178
17.309.2.2 check_eq() [2/2]	1178
17.309.2.3 cmp()	1179
17.309.2.4 eval_func_on_array() [1/3]	1179
17.309.2.5 eval_func_on_array() [2/3]	1179
17.309.2.6 eval_func_on_array() [3/3]	1179
17.309.2.7 operator<<()	1179
17.309.2.8 test_impl_base() [1/2]	1179
17.309.2.9 test_impl_base() [2/2]	1179
17.309.2.10 test_impl()	1179
17.309.2.11 main()	1179
17.310 test-solve.C File Reference	1180
17.310.1 Function Documentation	1180
17.310.1.1 check_solve()	1180
17.310.1.2 run_with_field()	1180
17.310.1.3 main()	1180
17.311 test-storage-transpose.C File Reference	1180
17.311.1 Function Documentation	1181
17.311.1.1 main()	1181
17.312 test-utils.h File Reference	1181
17.313 timer.h File Reference	1182
17.314 utils.h File Reference	1182
17.315 winograd.C File Reference	1182
17.315.1 Macro Definition Documentation	1183
17.315.1.1 DOUBLE_TO_FLOAT_CROSSOVER	1183
17.315.1.2 GFOPS	1183
17.315.2 Typedef Documentation	1183
17.315.2.1 TTimer	1183
17.315.3 Function Documentation	1183

17.315.3.1 <a href="#">balanced()</a> <sup>[1/2]</sup> . . . . .	1183
17.315.3.2 <a href="#">balanced()</a> <sup>[2/2]</sup> . . . . .	1183
17.315.3.3 <a href="#">main()</a> . . . . .	1183

<b><a href="#">Index</a></b>	<b>1185</b>
------------------------------	-------------

# Chapter 1

## FFLAS-FFPACK Documentation.

### 1.1 Introduction

FFLAS-FFPACK is a LGPL-2.1+ source code library for basic linear algebra operations over a finite field. It is inspired by BLAS interface (Basic Linear Algebra Subprograms) and the LAPACK library for numerical linear algebra, and shares part of their design. Yet it differs in many aspects due to the specificities of computing over a finite field:

- it is generic with respect to the finite field, so as to accomodate a large variety of field sizes and implementations;
- it is a pure source code library, to be included and compiled in the user's software. Its build system is only used for tests and benchmarks.

### 1.2 Goals

### 1.3 Design

### 1.4 Using FFLAS-FFPACK.

- [Copying and Licence](#).
- [Tutorial](#). This is a brief introduction to FFLAS-FFPACK capabilities.
- [Configuring and Installing FFLAS-FFPACK](#). Explains how to configure/install from sources or from the latest svn version.
- [Architecture of the library](#).. Describes how FFLAS-FFPACK is organized
- [Documentation for Users](#). If everything around is blue, then you are reading the lighter, user-oriented, documentation.
- [Documentation for Developers](#). If everything around is green, then you can get to everything (not necessarily yet) documented.

### 1.5 Contributing to fflas-ffpack, getting assistance.

Version

2.5.0



## Chapter 2

# Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Howver configuration process can be tweaked a lot. Configure looks for BLAS routines and [Givaro](#) library which are both mandatory dependencies. See the output of `./configure -help` for information about the LAPACK/BLAS discovering strategies.



## Chapter 3

# Copying and Licence

The FFLAS-FFPACK library is licensed under the terms of the GNU LGPL v2.1 or later.

See <https://www.gnu.org/licenses/lgpl-2.1.html>



## Chapter 4

# Tutorial

no doc.



## Chapter 5

# Architecture of the library.

no doc.



## Chapter 6

# Bug List

Global **DOUBLE\_TO\_FLOAT\_CROSSOVER**

to be benchmarked.

Global **FFLAS::details::pack\_lhs** (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global **FFLAS::details::pack\_rhs** (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global **FFLAS::fconvert** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX, const FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::fconvert** (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::finit** (const Field &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::finit** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fneg** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fneg** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fnegin** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fnegin** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::freduce** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::freduce** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fscale** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::fscaln** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fsquare** (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)

why double ?

Global **FFLAS::fswap** (const Field &F, const size\_t N, typename Field::Element\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)

use cblas\_dswap when double

Global **FFLAS::fswap** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_dswap when double

Global **FFLAS::ftrsm** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*B, const size\_t ldb)

$\alpha$  must be non zero.

Global **FFLAS::ftrsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)

$\alpha$  must be non zero.

Global **FFPACK::buildMatrix** (const Field &F, typename Field::ConstElement\_ptr E, typename Field::ConstElement\_ptr C, const size\_t lda, const size\_t \*B, const size\_t \*T, const size\_t me, const size\_t mc, const size\_t lambda, const size\_t mu)

is this :

Global **FFPACK::invert2** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t idx, int &nullity)

not tested.

Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename Field::Element alpha, const size\_t iters, RandIter &G)

test for incx equal

test for transpo

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size\_t iters, RandIter &G)

test for ldX equal

test for transpo

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size\_t iters, const int nbw, const bool par, RandIter &G)

test for ldX equal

test for transpo

Global **printvect** (std::ostream &o, vector< T > &vect)

does not belong here

## Chapter 7

# Bibliography

- Global **FFLAS::Protected::TRSMBound** (const Givaro::ModularBalanced< Element > &F) .  
Dumas Giorgi Pernet 06, arXiv:cs/0601133
- Global **FFPACK::LeadingSubmatrixRankProfiles** (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.
- Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive, const size\_t cutoff=\_\_FFLASFFPACK\_LUDIVINE\_THRESHOLD) .  
Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013  
• Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002
- Global **FFPACK::PLUQ** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Q) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013
- Global **FFPACK::productBruhatxTS** (const Field &Fi, size\_t N, size\_t s, size\_t r, size\_t t, const size\_t \*P, const size\_t \*Q, typename **Field::ConstElement\_ptr** Xu, size\_t ldu, size\_t NbBlocksU, const size\_t \*Ku, const size\_t \*Tu, const size\_t \*MU, typename **Field::ConstElement\_ptr** XI, size\_t ldl, size\_t NbBlocksL, const size\_t \*KI, const size\_t \*TI, const size\_t \*ML, typename **Field::Element\_ptr** B, size\_t ldb, const typename **Field::Element** beta, typename **Field::Element\_ptr** D, size\_t ldd) .  
Pernet C. and Storjohann A. *Time and space efficient generators for quasiseparable matrices*, JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010
- Global **FFPACK::Protected::GaussJordan** (const Field &F, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag) .  
Algorithm 2.8 of A. Storjohann Thesis 2000,  
• Algorithm 11 of Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Class **ftrsmLeftUpperNoTransNonUnit**< Element > .  
Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.



## Chapter 8

# Todo List

### File `debug.h`

we should put vector printing elsewhere.

Global `FFLAS::fadd` (const Field &F, const size\_t N, typename `Field::ConstElement_ptr` A, const size\_t incA, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` B, const size\_t incB, typename `Field::Element_ptr` C, const size\_t incC)

optimise here

Global `FFLAS::fassign` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

variant for triangular matrix

Global `FFLAS::fconvert` (const Field &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb)

check if n == lda

Global `FFLAS::fneg` (const Field &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fnegin` (const Field &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fscal` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::Protected::igemm` (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, const int64\_t beta, int64\_t \*C, size\_t ldc)

use primitive (no `Field()`) and specialise for int64.

Global `FFLAS::Protected::MatF2MatFI_Triangular` (const Field &F, Givaro::FloatDomain::Element\_ptr S, const size\_t lds, typename `Field::ConstElement_ptr` const E, const size\_t lde, const size\_t m, const size\_t n)

do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)

do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Q, const **FFPACK::FFPACK\_LU\_TAG** LuTag, const size\_t cutoff)  
std::swap ?

Global **FFPACK::Protected::RandomKrylovPrecond** (const PolRing &PR, std::list< typename PolRing::↵  
Element > &completedFactors, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const  
size\_t lda, size\_t &Nb, typename PolRing::Domain\_t::Element\_ptr &B, size\_t &ldb, typename PolRing↵  
::Domain\_t::Randlter &g, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
swap to save space ??  
don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

### Module **field**

biblio

Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename **Field::Element** alpha, const  
size\_t iters, Randlter &G)  
does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename  
**Field::Element** alpha, const typename **Field::Element** beta, const size\_t iters, Randlter &G)  
does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename  
**Field::Element** alpha, const typename **Field::Element** beta, const size\_t iters, const int nbw, const bool  
par, Randlter &G)  
does nbw actually do nbw recursive calls and then call blas (check ?) ?

### Module **MMalgos**

biblio

### Module **simd**

biblio

Global **test\_colechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG** LuTag,  
Randlter &G, bool par)  
check lda

Global **test\_det** (Field &F, size\_t n, int iter, Randlter &G)  
test with stride

Global **test\_redcochelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG**  
LuTag, Randlter &G, bool par)  
check lda

Global **test\_redrowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG**  
LuTag, Randlter &G, bool par)  
check lda

Global **test\_rowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG** LuTag,  
Randlter &G, bool par)  
check lda

## Chapter 9

# Module Index

### 9.1 Modules

Here is a list of all modules:

CHECKER . . . . .	45
Matrix Multiplication Algorithms . . . . .	45
SIMD wrapper . . . . .	46
FFLAS-FFPACK . . . . .	46
FFLAS . . . . .	45
Interfaces . . . . .	47
FFPACK . . . . .	46
FFLAS-FFPACK fields . . . . .	46
RNS . . . . .	47



## Chapter 10

# Namespace Index

### 10.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">FFLAS</a>	49
<a href="#">FFLAS::_ftranspose_impl</a>	203
<a href="#">FFLAS::BLAS3</a>	204
<a href="#">FFLAS::csr_hyb_details</a>	211
<a href="#">FFLAS::CuttingStrategy</a>	211
<a href="#">FFLAS::details</a>	212
<a href="#">FFLAS::details_spmv</a>	221
<a href="#">FFLAS::ElementCategories</a>	221
<a href="#">FFLAS::FieldCategories</a>	
Traits and categories will need to be placed in a proper file later	221
<a href="#">FFLAS::MMHelperAlgo</a>	222
<a href="#">FFLAS::ModeCategories</a>	
Specifies the mode of action for an algorithm w.r.t	222
<a href="#">FFLAS::ParSeqHelper</a>	
<a href="#">ParSeqHelper</a> for both fgemm and ftrsm	222
<a href="#">FFLAS::Protected</a>	223
<a href="#">FFLAS::sell_details</a>	238
<a href="#">FFLAS::sparse_details</a>	238
<a href="#">FFLAS::sparse_details_impl</a>	254
<a href="#">FFLAS::StrategyParameter</a>	296
<a href="#">FFLAS::StructureHelper</a>	
<a href="#">StructureHelper</a> for ftrsm	296
<a href="#">FFLAS::vectorised</a>	296
<a href="#">FFLAS::vectorised::unswitch</a>	303
<a href="#">FFPACK</a>	
<b>Finite Field PACK</b> Set of elimination based routines for dense linear algebra	305
<a href="#">FFPACK::Protected</a>	417
<a href="#">Givaro</a>	424
<a href="#">MKL_CONFIG</a>	424
<a href="#">RecInt</a>	424



## Chapter 11

# Hierarchical Index

### 11.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AlgoChooser< ModeT, ParSeq > . . . . .	427
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > . . . . .	427
ALL< v > . . . . .	427
ALL< false, v... > . . . . .	428
ALL< true, v... > . . . . .	428
ALL<> . . . . .	428
ArbitraryPrecIntTag . . . . .	428
AreEqual< X, Y > . . . . .	429
AreEqual< X, X > . . . . .	429
Argument . . . . .	429
associatedDelayedField< Field > . . . . .	430
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > . . . . .	430
associatedDelayedField< const Givaro::Modular< T, X > > . . . . .	431
associatedDelayedField< const Givaro::ModularBalanced< T > > . . . . .	431
associatedDelayedField< const Givaro::ZRing< T > > . . . . .	432
Auto . . . . .	432
Bench< Elt > . . . . .	432
Bini . . . . .	435
Block . . . . .	435
BlockTransposeSIMD< Field, Simd, > . . . . .	435
callLUdivine_small< Element > . . . . .	437
callLUdivine_small< double > . . . . .	437
callLUdivine_small< float > . . . . .	438
CharpolyFailed . . . . .	438
Checker_Empty< Field > . . . . .	438
CheckerImplem_charpoly< Field, Polynomial > . . . . .	439
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > . . . . .	440
CheckerImplem_Det< Field > . . . . .	441
CheckerImplem_fgemm< Field > . . . . .	442
CheckerImplem_ftsrsm< Field > . . . . .	443
CheckerImplem_invert< Field > . . . . .	444
CheckerImplem_PLUQ< Field > . . . . .	445
Classic . . . . .	446
Column . . . . .	446
CompactElement< Element > . . . . .	447

CompactElement< double > . . . . .	447
CompactElement< float > . . . . .	447
CompactElement< int16_t > . . . . .	447
CompactElement< int32_t > . . . . .	448
CompactElement< int64_t > . . . . .	448
compatible_data_type< Field > . . . . .	448
compatible_data_type< Givaro::ZRing< double > > . . . . .	449
compatible_data_type< Givaro::ZRing< float > > . . . . .	449
Compose< H1, H2 > . . . . .	449
Simd128_impl< true, true, false, 2 >::Converter . . . . .	451
Simd128_impl< true, true, false, 4 >::Converter . . . . .	451
Simd128_impl< true, true, false, 8 >::Converter . . . . .	451
Simd128_impl< true, true, true, 2 >::Converter . . . . .	452
Simd128_impl< true, true, true, 4 >::Converter . . . . .	452
Simd128_impl< true, true, true, 8 >::Converter . . . . .	453
Simd256_impl< true, false, true, 8 >::Converter . . . . .	453
Simd256_impl< true, true, false, 2 >::Converter . . . . .	453
Simd256_impl< true, true, false, 4 >::Converter . . . . .	454
Simd256_impl< true, true, false, 8 >::Converter . . . . .	454
Simd256_impl< true, true, true, 2 >::Converter . . . . .	455
Simd256_impl< true, true, true, 4 >::Converter . . . . .	455
Simd256_impl< true, true, true, 8 >::Converter . . . . .	455
Simd512_impl< true, true, false, 8 >::Converter . . . . .	456
Simd512_impl< true, true, true, 8 >::Converter . . . . .	456
ConvertTo< T > . . . . .	457
Coo< ValT, IdxT > . . . . .	457
Coo< Field > . . . . .	459
Coo< ValT, IdxT > . . . . .	460
CooMat< Field > . . . . .	462
CooMat< FFPACK::RNSInteger > . . . . .	462
count_nonconst_lvalue_reference< T > . . . . .	463
count_nonconst_lvalue_reference< const T &, O... > . . . . .	463
count_nonconst_lvalue_reference< T &, O... > . . . . .	463
count_nonconst_lvalue_reference< T, O... > . . . . .	463
count_nonconst_lvalue_reference<> . . . . .	464
CsrMat< Field > . . . . .	464
CsrMat< FFPACK::RNSInteger > . . . . .	464
DefaultBoundedTag . . . . .	465
DefaultTag . . . . .	465
DelayedTag . . . . .	465
DivideAndConquer . . . . .	465
ElementTraits< Element > . . . . .	466
ElementTraits< double > . . . . .	466
ElementTraits< FFPACK::rns_double_elt > . . . . .	466
ElementTraits< float > . . . . .	467
ElementTraits< Givaro::Integer > . . . . .	467
ElementTraits< int16_t > . . . . .	467
ElementTraits< int32_t > . . . . .	468
ElementTraits< int64_t > . . . . .	468
ElementTraits< int8_t > . . . . .	468
ElementTraits< Reclnt::rint< K > > . . . . .	469
ElementTraits< Reclnt::rmint< K, MG > > . . . . .	469
ElementTraits< Reclnt::ruint< K > > . . . . .	469
ElementTraits< uint16_t > . . . . .	470
ElementTraits< uint32_t > . . . . .	470
ElementTraits< uint64_t > . . . . .	470
ElementTraits< uint8_t > . . . . .	471
EllMat< Field > . . . . .	471

EllMat< FFPACK::RNSInteger > . . . . .	471
Failure . . . . .	472
FailureCharpolyCheck . . . . .	474
FailureDetCheck . . . . .	474
FailureFgemmCheck . . . . .	474
FailureInvertCheck . . . . .	474
FailurePLUQCheck . . . . .	474
FailureTrsmCheck . . . . .	474
false_type	
isSparseMatrix< Field, M > . . . . .	511
isSparseMatrixMKLFormat< F, M > . . . . .	515
isSparseMatrixSimdFormat< F, M > . . . . .	515
isZOSparseMatrix< F, M > . . . . .	515
support_fast_mod< T > . . . . .	808
support_simd< T > . . . . .	809
support_simd_add< T > . . . . .	810
support_simd_mod< T > . . . . .	810
FieldSimd< _Field > . . . . .	474
FieldTraits< Field > . . . . .	481
FieldTraits< FFPACK::RNSInteger< T > > . . . . .	481
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .	482
FieldTraits< Givaro::Modular< Element > > . . . . .	482
FieldTraits< Givaro::ModularBalanced< Element > > . . . . .	483
FieldTraits< Givaro::ZRing< double > > . . . . .	483
FieldTraits< Givaro::ZRing< float > > . . . . .	484
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	484
FieldTraits< Givaro::ZRing< int16_t > > . . . . .	485
FieldTraits< Givaro::ZRing< int32_t > > . . . . .	486
FieldTraits< Givaro::ZRing< int64_t > > . . . . .	486
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . .	487
FieldTraits< Givaro::ZRing< uint16_t > > . . . . .	487
FieldTraits< Givaro::ZRing< uint32_t > > . . . . .	488
FieldTraits< Givaro::ZRing< uint64_t > > . . . . .	488
Fixed . . . . .	489
FixedPreclntTag . . . . .	489
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >← ::FloatingPointTestDistribution . . . . .	489
ForStrategy1D< blocksize_t, Cut, Param > . . . . .	490
ForStrategy2D< blocksize_t, Cut, Param > . . . . .	492
ftmmLeftLowerNoTransNonUnit< Element > . . . . .	495
ftmmLeftLowerNoTransUnit< Element > . . . . .	496
ftmmLeftLowerTransNonUnit< Element > . . . . .	496
ftmmLeftLowerTransUnit< Element > . . . . .	496
ftmmLeftUpperNoTransNonUnit< Element > . . . . .	496
ftmmLeftUpperNoTransUnit< Element > . . . . .	496
ftmmLeftUpperTransNonUnit< Element > . . . . .	496
ftmmLeftUpperTransUnit< Element > . . . . .	496
ftmmRightLowerNoTransNonUnit< Element > . . . . .	496
ftmmRightLowerNoTransUnit< Element > . . . . .	497
ftmmRightLowerTransNonUnit< Element > . . . . .	497
ftmmRightLowerTransUnit< Element > . . . . .	497
ftmmRightUpperNoTransNonUnit< Element > . . . . .	497
ftmmRightUpperNoTransUnit< Element > . . . . .	497
ftmmRightUpperTransNonUnit< Element > . . . . .	497
ftmmRightUpperTransUnit< Element > . . . . .	497
ftsmLeftLowerNoTransNonUnit< Element > . . . . .	497
ftsmLeftLowerNoTransUnit< Element > . . . . .	498
ftsmLeftLowerTransNonUnit< Element > . . . . .	498

ftsmLeftLowerTransUnit< Element > . . . . .	498
ftsmLeftUpperNoTransNonUnit< Element > . . . . .	498
ftsmLeftUpperNoTransUnit< Element > . . . . .	498
ftsmLeftUpperTransNonUnit< Element > . . . . .	499
ftsmLeftUpperTransUnit< Element > . . . . .	499
ftsmRightLowerNoTransNonUnit< Element > . . . . .	499
ftsmRightLowerNoTransUnit< Element > . . . . .	499
ftsmRightLowerTransNonUnit< Element > . . . . .	499
ftsmRightLowerTransUnit< Element > . . . . .	499
ftsmRightUpperNoTransNonUnit< Element > . . . . .	499
ftsmRightUpperNoTransUnit< Element > . . . . .	499
ftsmRightUpperTransNonUnit< Element > . . . . .	500
ftsmRightUpperTransUnit< Element > . . . . .	500
GenericTag . . . . .	500
GenericTag . . . . .	500
Grain . . . . .	500
has_minus_eq_impl< C > . . . . .	500
has_minus_impl< C > . . . . .	501
has_mul_eq_impl< C > . . . . .	501
has_mul_impl< C > . . . . .	501
has_operation< T > . . . . .	502
has_plus_eq_impl< C > . . . . .	502
has_plus_impl< C > . . . . .	502
HelperFlag . . . . .	503
HelperMod< Field, ElementTraits > . . . . .	504
HelperMod< Field, ElementCategories::MachineIntTag > . . . . .	504
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > . . . . .	505
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > . . . . .	505
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > . . . . .	506
Hybrid . . . . .	507
Info . . . . .	507
Info . . . . .	508
is_all_same< Args > . . . . .	510
is_all_same< T, Args... > . . . . .	510
is_all_same<> . . . . .	510
is_simd< T > . . . . .	511
Iterative . . . . .	517
LazyTag . . . . .	517
limits< T > . . . . .	518
limits< char > . . . . .	518
limits< double > . . . . .	518
limits< float > . . . . .	519
limits< Givaro::Integer > . . . . .	520
limits< int > . . . . .	521
limits< long > . . . . .	521
limits< long long > . . . . .	522
limits< Reclnt::rint< K > > . . . . .	523
limits< Reclnt::ruint< K > > . . . . .	523
limits< short int > . . . . .	524
limits< signed char > . . . . .	525
limits< unsigned char > . . . . .	525
limits< unsigned int > . . . . .	526
limits< unsigned long > . . . . .	527
limits< unsigned long long > . . . . .	528
limits< unsigned short int > . . . . .	528
MachineFloatTag . . . . .	529
MachineIntTag . . . . .	529
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > . . . . .	530

MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	535
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	537
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > . . . . .	539
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > . . . . .	541
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	543
ModeTraits< Field > . . . . .	544
ModeTraits< Givaro::Modular< Element, Compute > > . . . . .	545
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > . . . . .	545
ModeTraits< Givaro::Modular< int16_t, Compute > > . . . . .	546
ModeTraits< Givaro::Modular< int32_t, Compute > > . . . . .	546
ModeTraits< Givaro::Modular< int64_t, uint64_t > > . . . . .	546
ModeTraits< Givaro::Modular< int8_t, Compute > > . . . . .	547
ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > > . . . . .	547
ModeTraits< Givaro::Modular< uint16_t, Compute > > . . . . .	547
ModeTraits< Givaro::Modular< uint32_t, Compute > > . . . . .	548
ModeTraits< Givaro::Modular< uint8_t, Compute > > . . . . .	548
ModeTraits< Givaro::ModularBalanced< Element > > . . . . .	549
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . . .	549
ModeTraits< Givaro::ModularBalanced< int16_t > > . . . . .	549
ModeTraits< Givaro::ModularBalanced< int32_t > > . . . . .	550
ModeTraits< Givaro::ModularBalanced< int8_t > > . . . . .	550
ModeTraits< Givaro::Montgomery< T > > . . . . .	550
ModeTraits< Givaro::ZRing< double > > . . . . .	551
ModeTraits< Givaro::ZRing< float > > . . . . .	551
ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	551
ModularBalanced< T > . . . . .	552
ModularTag . . . . .	552
Montgomery< T > . . . . .	552
need_field_characteristic< Field > . . . . .	552
need_field_characteristic< Givaro::Modular< Field > > . . . . .	552
need_field_characteristic< Givaro::ModularBalanced< Field > > . . . . .	553
NoSimd< T > . . . . .	553
Parallel< C, P > . . . . .	555
readMyMachineType< Field, T > . . . . .	558
readMyMachineType< Field, mpz_t > . . . . .	559
Recursive . . . . .	560
Recursive . . . . .	560
rint< K > . . . . .	560
rns_double . . . . .	560
rns_double_elt . . . . .	565
rns_double_elt_cstptr . . . . .	567
rns_double_elt_ptr . . . . .	570
rns_double_extended . . . . .	573
RNSElementTag . . . . .	577
RNSInteger< RNS > . . . . .	578
RNSInteger< FFPACK::rns_double > . . . . .	578
RNSIntegerMod< RNS > . . . . .	581
RNSIntegerMod< FFPACK::rns_double > . . . . .	581
rnsRandIter< RNS > . . . . .	588
RNSInteger< RNS >::RandIter . . . . .	556
RNSIntegerMod< RNS >::RandIter . . . . .	557
Row . . . . .	589
ruint< K > . . . . .	589
ScalFunctionsBase< Element, Enable > . . . . .	594
ScalFunctions< Element > . . . . .	589
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > . . . . .	595

ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > . . . . .	596
Sequential . . . . .	599
Simd128_impl< ArithType, Int, Signed, Size > . . . . .	600
Simd128_impl< true, false, true, 4 > . . . . .	600
Simd128_impl< true, false, true, 8 > . . . . .	600
Simd128i_base . . . . .	661
Simd128_impl< true, true, true, 2 > . . . . .	631
Simd128_impl< true, true, false, 2 > . . . . .	600
Simd128_impl< true, true, true, 4 > . . . . .	641
Simd128_impl< true, true, false, 4 > . . . . .	610
Simd128_impl< true, true, true, 8 > . . . . .	651
Simd128_impl< true, true, false, 8 > . . . . .	620
Simd256_impl< ArithType, Int, Signed, Size > . . . . .	663
Simd256fp_base . . . . .	750
Simd256_impl< true, false, true, 4 > . . . . .	663
Simd256_impl< true, false, true, 8 > . . . . .	663
Simd256i_base . . . . .	751
Simd256_impl< true, true, true, 2 > . . . . .	711
Simd256_impl< true, true, false, 2 > . . . . .	672
Simd256_impl< true, true, true, 4 > . . . . .	721
Simd256_impl< true, true, false, 4 > . . . . .	682
Simd256_impl< true, true, false, 4 > . . . . .	682
Simd256_impl< true, true, true, 8 > . . . . .	740
Simd256_impl< true, true, false, 8 > . . . . .	701
Simd512_impl< ArithType, Int, Signed, Size > . . . . .	751
Simd512_impl< true, false, true, 4 > . . . . .	751
Simd512_impl< true, false, true, 8 > . . . . .	751
Simd512i_base . . . . .	781
Simd256_impl< true, true, true, 4 > . . . . .	721
Simd512_impl< true, true, true, 8 > . . . . .	770
Simd512_impl< true, true, false, 8 > . . . . .	759
SimdChooser< T, bool, bool > . . . . .	782
SimdChooser< T, false, b > . . . . .	782
SimdChooser< T, true, false > . . . . .	783
SimdChooser< T, true, true > . . . . .	783
simdToType< T > . . . . .	783
Single . . . . .	783
Sparse< Field, SparseMatrix_t, IdxT, PtrT > . . . . .	783
Sparse< _Field, SparseMatrix_t::COO > . . . . .	784
Sparse< _Field, SparseMatrix_t::COO_ZO > . . . . .	785
Sparse< _Field, SparseMatrix_t::CSR > . . . . .	787
Sparse< _Field, SparseMatrix_t::CSR_ZO > . . . . .	790
Sparse< _Field, SparseMatrix_t::CSR_HYB > . . . . .	789
Sparse< _Field, SparseMatrix_t::ELL > . . . . .	792
Sparse< _Field, SparseMatrix_t::ELL_ZO > . . . . .	797
Sparse< _Field, SparseMatrix_t::ELL_simd > . . . . .	794
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > . . . . .	795
Sparse< _Field, SparseMatrix_t::HYB_ZO > . . . . .	799
Sparse< _Field, SparseMatrix_t::SELL > . . . . .	800
Sparse< _Field, SparseMatrix_t::SELL_ZO > . . . . .	802
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int32_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int64_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int32_t > . . . . .	783

Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int64_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int32_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int64_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int32_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int64_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int32_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int64_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int16_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int32_t > . . . . .	783
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int64_t > . . . . .	783
SpMat< Field, flag > . . . . .	804
StatsMatrix . . . . .	805
Test< Elt > . . . . .	810
TestOneMethod< Simd > . . . . .	813
tfn_minus . . . . .	816
tfn_minus_eq . . . . .	816
tfn_mul . . . . .	817
tfn_mul_eq . . . . .	817
tfn_plus . . . . .	817
tfn_plus_eq . . . . .	818
Threads . . . . .	818
ThreeD . . . . .	818
ThreeDAdaptive . . . . .	818
ThreeDInPlace . . . . .	818
TRSMHelper< ReclterTrait, ParSeqTrait > . . . . .	819
true_type	
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > . . . . .	511
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . . .	512
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > . . . . .	512
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > . . . . .	512
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . . .	513
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > . . . . .	513
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . . .	514
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > . . . . .	513
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . . .	513
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > . . . . .	514
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > . . . . .	514
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . . .	515
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . . .	516
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . . .	516
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . . .	517
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . . .	516
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . . .	517
support_fast_mod< double > . . . . .	808
support_fast_mod< float > . . . . .	809
support_fast_mod< int64_t > . . . . .	809
TwoD . . . . .	820
TwoDAdaptive . . . . .	820
UnparametricTag . . . . .	820
width< T > . . . . .	820
width< double > . . . . .	821
width< float > . . . . .	821
Winograd . . . . .	821
WinogradPar . . . . .	821



## Chapter 12

# Data Structure Index

### 12.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">AlgoChooser&lt; ModeT, ParSeq &gt;</a>	427
<a href="#">AlgoChooser&lt; ModeCategories::ConvertTo&lt; ElementCategories::RNSElementTag &gt;, ParSeq &gt;</a>	427
<a href="#">ALL&lt; v &gt;</a>	427
<a href="#">ALL&lt; false, v... &gt;</a>	428
<a href="#">ALL&lt; true, v... &gt;</a>	428
<a href="#">ALL&lt;&gt;</a>	428
<a href="#">ArbitraryPrecIntTag</a>	
Arbitrary precision integers: GMP	428
<a href="#">AreEqual&lt; X, Y &gt;</a>	429
<a href="#">AreEqual&lt; X, X &gt;</a>	429
<a href="#">Argument</a>	429
<a href="#">associatedDelayedField&lt; Field &gt;</a>	430
<a href="#">associatedDelayedField&lt; const FFPACK::RNSIntegerMod&lt; RNS &gt; &gt;</a>	430
<a href="#">associatedDelayedField&lt; const Givaro::Modular&lt; T, X &gt; &gt;</a>	431
<a href="#">associatedDelayedField&lt; const Givaro::ModularBalanced&lt; T &gt; &gt;</a>	431
<a href="#">associatedDelayedField&lt; const Givaro::ZRing&lt; T &gt; &gt;</a>	432
<a href="#">Auto</a>	432
<a href="#">Bench&lt; Elt &gt;</a>	432
<a href="#">Bini</a>	435
<a href="#">Block</a>	435
<a href="#">BlockTransposeSIMD&lt; Field, Simd, &gt;</a>	435
<a href="#">callLUdivine_small&lt; Element &gt;</a>	437
<a href="#">callLUdivine_small&lt; double &gt;</a>	437
<a href="#">callLUdivine_small&lt; float &gt;</a>	438
<a href="#">CharpolyFailed</a>	438
<a href="#">Checker_Empty&lt; Field &gt;</a>	438
<a href="#">CheckerImplem_charpoly&lt; Field, Polynomial &gt;</a>	439
<a href="#">CheckerImplem_charpoly&lt; Givaro::ZRing&lt; Givaro::Integer &gt;, Polynomial &gt;</a>	440
<a href="#">CheckerImplem_Det&lt; Field &gt;</a>	441
<a href="#">CheckerImplem_fgemm&lt; Field &gt;</a>	442
<a href="#">CheckerImplem_ftdsm&lt; Field &gt;</a>	443
<a href="#">CheckerImplem_invert&lt; Field &gt;</a>	444
<a href="#">CheckerImplem_PLUQ&lt; Field &gt;</a>	445
<a href="#">Classic</a>	446
<a href="#">Column</a>	446

<a href="#">CompactElement&lt; Element &gt;</a>	<a href="#">447</a>
<a href="#">CompactElement&lt; double &gt;</a>	<a href="#">447</a>
<a href="#">CompactElement&lt; float &gt;</a>	<a href="#">447</a>
<a href="#">CompactElement&lt; int16_t &gt;</a>	<a href="#">447</a>
<a href="#">CompactElement&lt; int32_t &gt;</a>	<a href="#">448</a>
<a href="#">CompactElement&lt; int64_t &gt;</a>	<a href="#">448</a>
<a href="#">compatible_data_type&lt; Field &gt;</a>	<a href="#">448</a>
<a href="#">compatible_data_type&lt; Givaro::ZRing&lt; double &gt; &gt;</a>	<a href="#">449</a>
<a href="#">compatible_data_type&lt; Givaro::ZRing&lt; float &gt; &gt;</a>	<a href="#">449</a>
<a href="#">Compose&lt; H1, H2 &gt;</a>	<a href="#">449</a>
<a href="#">Simd128_impl&lt; true, true, false, 2 &gt;::Converter</a>	<a href="#">451</a>
<a href="#">Simd128_impl&lt; true, true, false, 4 &gt;::Converter</a>	<a href="#">451</a>
<a href="#">Simd128_impl&lt; true, true, false, 8 &gt;::Converter</a>	<a href="#">451</a>
<a href="#">Simd128_impl&lt; true, true, true, 2 &gt;::Converter</a>	<a href="#">452</a>
<a href="#">Simd128_impl&lt; true, true, true, 4 &gt;::Converter</a>	<a href="#">452</a>
<a href="#">Simd128_impl&lt; true, true, true, 8 &gt;::Converter</a>	<a href="#">453</a>
<a href="#">Simd256_impl&lt; true, false, true, 8 &gt;::Converter</a>	<a href="#">453</a>
<a href="#">Simd256_impl&lt; true, true, false, 2 &gt;::Converter</a>	<a href="#">453</a>
<a href="#">Simd256_impl&lt; true, true, false, 4 &gt;::Converter</a>	<a href="#">454</a>
<a href="#">Simd256_impl&lt; true, true, false, 8 &gt;::Converter</a>	<a href="#">454</a>
<a href="#">Simd256_impl&lt; true, true, true, 2 &gt;::Converter</a>	<a href="#">455</a>
<a href="#">Simd256_impl&lt; true, true, true, 4 &gt;::Converter</a>	<a href="#">455</a>
<a href="#">Simd256_impl&lt; true, true, true, 8 &gt;::Converter</a>	<a href="#">455</a>
<a href="#">Simd512_impl&lt; true, true, false, 8 &gt;::Converter</a>	<a href="#">456</a>
<a href="#">Simd512_impl&lt; true, true, true, 8 &gt;::Converter</a>	<a href="#">456</a>
<a href="#">ConvertTo&lt; T &gt;</a>	
Force conversion to appropriate element type of ElementCategory T	<a href="#">457</a>
<a href="#">Coo&lt; ValT, IdxT &gt;</a>	<a href="#">457</a>
<a href="#">Coo&lt; Field &gt;</a>	<a href="#">459</a>
<a href="#">Coo&lt; ValT, IdxT &gt;</a>	<a href="#">460</a>
<a href="#">CooMat&lt; Field &gt;</a>	<a href="#">462</a>
<a href="#">count_nonconst_lvalue_reference&lt; T &gt;</a>	<a href="#">463</a>
<a href="#">count_nonconst_lvalue_reference&lt; const T &amp;, O... &gt;</a>	<a href="#">463</a>
<a href="#">count_nonconst_lvalue_reference&lt; T &amp;, O... &gt;</a>	<a href="#">463</a>
<a href="#">count_nonconst_lvalue_reference&lt; T, O... &gt;</a>	<a href="#">463</a>
<a href="#">count_nonconst_lvalue_reference&lt;&gt;</a>	<a href="#">464</a>
<a href="#">CsrMat&lt; Field &gt;</a>	<a href="#">464</a>
<a href="#">DefaultBoundedTag</a>	
Use standard field operations, but keeps track of bounds on input and output	<a href="#">465</a>
<a href="#">DefaultTag</a>	
No specific mode of action: use standard field operations	<a href="#">465</a>
<a href="#">DelayedTag</a>	
Performs field operations with delayed mod reductions. Ensures result is reduced	<a href="#">465</a>
<a href="#">DivideAndConquer</a>	<a href="#">465</a>
<a href="#">ElementTraits&lt; Element &gt;</a>	
<a href="#">ElementTraits</a>	<a href="#">466</a>
<a href="#">ElementTraits&lt; double &gt;</a>	<a href="#">466</a>
<a href="#">ElementTraits&lt; FFPACK::rns_double_elt &gt;</a>	<a href="#">466</a>
<a href="#">ElementTraits&lt; float &gt;</a>	<a href="#">467</a>
<a href="#">ElementTraits&lt; Givaro::Integer &gt;</a>	<a href="#">467</a>
<a href="#">ElementTraits&lt; int16_t &gt;</a>	<a href="#">467</a>
<a href="#">ElementTraits&lt; int32_t &gt;</a>	<a href="#">468</a>
<a href="#">ElementTraits&lt; int64_t &gt;</a>	<a href="#">468</a>
<a href="#">ElementTraits&lt; int8_t &gt;</a>	<a href="#">468</a>
<a href="#">ElementTraits&lt; Reclnt::rint&lt; K &gt; &gt;</a>	<a href="#">469</a>
<a href="#">ElementTraits&lt; Reclnt::rmint&lt; K, MG &gt; &gt;</a>	<a href="#">469</a>
<a href="#">ElementTraits&lt; Reclnt::ruint&lt; K &gt; &gt;</a>	<a href="#">469</a>
<a href="#">ElementTraits&lt; uint16_t &gt;</a>	<a href="#">470</a>

ElementTraits< uint32_t > . . . . .	470
ElementTraits< uint64_t > . . . . .	470
ElementTraits< uint8_t > . . . . .	471
ElMat< Field > . . . . .	471
Failure	
A precondition failed . . . . .	472
FailureCharpolyCheck . . . . .	474
FailureDetCheck . . . . .	474
FailureFgemmCheck . . . . .	474
FailureInvertCheck . . . . .	474
FailurePLUQCheck . . . . .	474
FailureTrsmCheck . . . . .	474
FieldSimd< _Field > . . . . .	474
FieldTraits< Field >	
FieldTrait . . . . .	481
FieldTraits< FFPACK::RNSInteger< T > > . . . . .	481
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .	482
FieldTraits< Givaro::Modular< Element > > . . . . .	482
FieldTraits< Givaro::ModularBalanced< Element > > . . . . .	483
FieldTraits< Givaro::ZRing< double > > . . . . .	483
FieldTraits< Givaro::ZRing< float > > . . . . .	484
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	484
FieldTraits< Givaro::ZRing< int16_t > > . . . . .	485
FieldTraits< Givaro::ZRing< int32_t > > . . . . .	486
FieldTraits< Givaro::ZRing< int64_t > > . . . . .	486
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . .	487
FieldTraits< Givaro::ZRing< uint16_t > > . . . . .	487
FieldTraits< Givaro::ZRing< uint32_t > > . . . . .	488
FieldTraits< Givaro::ZRing< uint64_t > > . . . . .	488
Fixed . . . . .	489
FixedPrecIntTag	
Fixed precision integers above machine precision: Givaro::reclnt . . . . .	489
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution	
489	
ForStrategy1D< blocksize_t, Cut, Param > . . . . .	490
ForStrategy2D< blocksize_t, Cut, Param > . . . . .	492
ftmmLeftLowerNoTransNonUnit< Element > . . . . .	495
ftmmLeftLowerNoTransUnit< Element > . . . . .	496
ftmmLeftLowerTransNonUnit< Element > . . . . .	496
ftmmLeftLowerTransUnit< Element > . . . . .	496
ftmmLeftUpperNoTransNonUnit< Element > . . . . .	496
ftmmLeftUpperNoTransUnit< Element > . . . . .	496
ftmmLeftUpperTransNonUnit< Element > . . . . .	496
ftmmLeftUpperTransUnit< Element > . . . . .	496
ftmmRightLowerNoTransNonUnit< Element > . . . . .	496
ftmmRightLowerNoTransUnit< Element > . . . . .	497
ftmmRightLowerTransNonUnit< Element > . . . . .	497
ftmmRightLowerTransUnit< Element > . . . . .	497
ftmmRightUpperNoTransNonUnit< Element > . . . . .	497
ftmmRightUpperNoTransUnit< Element > . . . . .	497
ftmmRightUpperTransNonUnit< Element > . . . . .	497
ftmmRightUpperTransUnit< Element > . . . . .	497
ftsmLeftLowerNoTransNonUnit< Element > . . . . .	497
ftsmLeftLowerNoTransUnit< Element > . . . . .	498
ftsmLeftLowerTransNonUnit< Element > . . . . .	498
ftsmLeftLowerTransUnit< Element > . . . . .	498
ftsmLeftUpperNoTransNonUnit< Element > . . . . .	498
Computes the maximal size for delaying the modular reduction in a triangular system resolution	498

<a href="#">ftrsmLeftUpperNoTransUnit&lt; Element &gt;</a>	498
<a href="#">ftrsmLeftUpperTransNonUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmLeftUpperTransUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightLowerNoTransNonUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightLowerNoTransUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightLowerTransNonUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightLowerTransUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightUpperNoTransNonUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightUpperNoTransUnit&lt; Element &gt;</a>	499
<a href="#">ftrsmRightUpperTransNonUnit&lt; Element &gt;</a>	500
<a href="#">ftrsmRightUpperTransUnit&lt; Element &gt;</a>	500
<a href="#">GenericTag</a>	
Default is generic	500
<a href="#">GenericTag</a>	
Generic ring	500
<a href="#">Grain</a>	500
<a href="#">has_minus_eq_impl&lt; C &gt;</a>	500
<a href="#">has_minus_impl&lt; C &gt;</a>	501
<a href="#">has_mul_eq_impl&lt; C &gt;</a>	501
<a href="#">has_mul_impl&lt; C &gt;</a>	501
<a href="#">has_operation&lt; T &gt;</a>	502
<a href="#">has_plus_eq_impl&lt; C &gt;</a>	502
<a href="#">has_plus_impl&lt; C &gt;</a>	502
<a href="#">HelperFlag</a>	503
<a href="#">HelperMod&lt; Field, ElementTraits &gt;</a>	504
<a href="#">HelperMod&lt; Field, ElementCategories::MachineIntTag &gt;</a>	504
<a href="#">HelperMod&lt; Field, FFLAS::ElementCategories::ArbitraryPrecIntTag &gt;</a>	505
<a href="#">HelperMod&lt; Field, FFLAS::ElementCategories::FixedPrecIntTag &gt;</a>	505
<a href="#">HelperMod&lt; Field, FFLAS::ElementCategories::MachineFloatTag &gt;</a>	506
<a href="#">Hybrid</a>	507
<a href="#">Info</a>	507
<a href="#">Info</a>	508
<a href="#">is_all_same&lt; Args &gt;</a>	510
<a href="#">is_all_same&lt; T, Args... &gt;</a>	510
<a href="#">is_all_same&lt;&gt;</a>	510
<a href="#">is_simd&lt; T &gt;</a>	511
<a href="#">isSparseMatrix&lt; Field, M &gt;</a>	511
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::COO &gt;&gt;</a>	511
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::COO_ZO &gt;&gt;</a>	512
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::CSR &gt;&gt;</a>	512
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::CSR_HYB &gt;&gt;</a>	512
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::CSR_ZO &gt;&gt;</a>	513
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL &gt;&gt;</a>	513
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL_simd &gt;&gt;</a>	513
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL_simd_ZO &gt;&gt;</a>	513
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL_ZO &gt;&gt;</a>	514
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::HYB_ZO &gt;&gt;</a>	514
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::SELL &gt;&gt;</a>	514
<a href="#">isSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::SELL_ZO &gt;&gt;</a>	515
<a href="#">isSparseMatrixMKLFormat&lt; F, M &gt;</a>	515
<a href="#">isSparseMatrixSimdFormat&lt; F, M &gt;</a>	515
<a href="#">isZOSparseMatrix&lt; F, M &gt;</a>	515
<a href="#">isZOSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::COO_ZO &gt;&gt;</a>	516
<a href="#">isZOSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::CSR_ZO &gt;&gt;</a>	516
<a href="#">isZOSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL_simd_ZO &gt;&gt;</a>	516
<a href="#">isZOSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::ELL_ZO &gt;&gt;</a>	517
<a href="#">isZOSparseMatrix&lt; Field, Sparse&lt; Field, SparseMatrix_t::SELL_ZO &gt;&gt;</a>	517
<a href="#">Iterative</a>	517

## LazyTag

Performs field operations with delayed mod only when necessary. Result may not be reduced	517
limits< T >	518
limits< char >	518
limits< double >	518
limits< float >	519
limits< Givaro::Integer >	520
limits< int >	521
limits< long >	521
limits< long long >	522
limits< RecInt::rint< K > >	523
limits< RecInt::ruint< K > >	523
limits< short int >	524
limits< signed char >	525
limits< unsigned char >	525
limits< unsigned int >	526
limits< unsigned long >	527
limits< unsigned long long >	528
limits< unsigned short int >	528
MachineFloatTag	
Float or double	529
MachineIntTag	
Short, int, long, long long, and unsigned variants	529
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >	530
MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	535
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	537
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >	539
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >	541
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	
FGEMM Helper for Default and ConvertTo modes of operation	543
ModeTraits< Field >	
ModeTraits	544
ModeTraits< Givaro::Modular< Element, Compute > >	545
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >	545
ModeTraits< Givaro::Modular< int16_t, Compute > >	546
ModeTraits< Givaro::Modular< int32_t, Compute > >	546
ModeTraits< Givaro::Modular< int64_t, uint64_t > >	546
ModeTraits< Givaro::Modular< int8_t, Compute > >	547
ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >	547
ModeTraits< Givaro::Modular< uint16_t, Compute > >	547
ModeTraits< Givaro::Modular< uint32_t, Compute > >	548
ModeTraits< Givaro::Modular< uint8_t, Compute > >	548
ModeTraits< Givaro::ModularBalanced< Element > >	549
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >	549
ModeTraits< Givaro::ModularBalanced< int16_t > >	549
ModeTraits< Givaro::ModularBalanced< int32_t > >	550
ModeTraits< Givaro::ModularBalanced< int8_t > >	550
ModeTraits< Givaro::Montgomery< T > >	550
ModeTraits< Givaro::ZRing< double > >	551
ModeTraits< Givaro::ZRing< float > >	551
ModeTraits< Givaro::ZRing< Givaro::Integer > >	551
ModularBalanced< T >	552
ModularTag	
This is a modular field like e.g. Modular<T> or ModularBalanced<T>	552
Montgomery< T >	552
need_field_characteristic< Field >	552
need_field_characteristic< Givaro::Modular< Field > >	552

<a href="#">need_field_characteristic&lt; Givaro::ModularBalanced&lt; Field &gt; &gt;</a>	553
<a href="#">NoSimd&lt; T &gt;</a>	553
<a href="#">Parallel&lt; C, P &gt;</a>	555
<a href="#">RNSInteger&lt; RNS &gt;::RandIter</a>	556
<a href="#">RNSIntegerMod&lt; RNS &gt;::RandIter</a>	557
<a href="#">readMyMachineType&lt; Field, T &gt;</a>	558
<a href="#">readMyMachineType&lt; Field, mpz_t &gt;</a>	559
<a href="#">Recursive</a>	560
<a href="#">Recursive</a>	560
<a href="#">rint&lt; K &gt;</a>	560
<a href="#">rns_double</a>	560
<a href="#">rns_double_elt</a>	565
<a href="#">rns_double_elt_cstptr</a>	567
<a href="#">rns_double_elt_ptr</a>	570
<a href="#">rns_double_extended</a>	573
<a href="#">RNSElementTag</a>	
Representation in a Residue Number System	577
<a href="#">RNSInteger&lt; RNS &gt;</a>	578
<a href="#">RNSIntegerMod&lt; RNS &gt;</a>	581
<a href="#">rnsRandIter&lt; RNS &gt;</a>	588
<a href="#">Row</a>	589
<a href="#">ruint&lt; K &gt;</a>	589
<a href="#">ScalFunctions&lt; Element &gt;</a>	589
<a href="#">ScalFunctionsBase&lt; Element, Enable &gt;</a>	594
<a href="#">ScalFunctionsBase&lt; Element, typename enable_if&lt; is_floating_point&lt; Element &gt;::value &gt;::type &gt;</a>	595
<a href="#">ScalFunctionsBase&lt; Element, typename enable_if&lt; is_integral&lt; Element &gt;::value &gt;::type &gt;</a>	596
<a href="#">Sequential</a>	599
<a href="#">Simd128_impl&lt; ArithType, Int, Signed, Size &gt;</a>	600
<a href="#">Simd128_impl&lt; true, false, true, 4 &gt;</a>	600
<a href="#">Simd128_impl&lt; true, false, true, 8 &gt;</a>	600
<a href="#">Simd128_impl&lt; true, true, false, 2 &gt;</a>	600
<a href="#">Simd128_impl&lt; true, true, false, 4 &gt;</a>	610
<a href="#">Simd128_impl&lt; true, true, false, 8 &gt;</a>	620
<a href="#">Simd128_impl&lt; true, true, true, 2 &gt;</a>	631
<a href="#">Simd128_impl&lt; true, true, true, 4 &gt;</a>	641
<a href="#">Simd128_impl&lt; true, true, true, 8 &gt;</a>	651
<a href="#">Simd128i_base</a>	661
<a href="#">Simd256_impl&lt; ArithType, Int, Signed, Size &gt;</a>	663
<a href="#">Simd256_impl&lt; true, false, true, 4 &gt;</a>	663
<a href="#">Simd256_impl&lt; true, false, true, 8 &gt;</a>	663
<a href="#">Simd256_impl&lt; true, true, false, 2 &gt;</a>	672
<a href="#">Simd256_impl&lt; true, true, false, 4 &gt;</a>	682
<a href="#">Simd256_impl&lt; true, true, false, 8 &gt;</a>	701
<a href="#">Simd256_impl&lt; true, true, true, 2 &gt;</a>	711
<a href="#">Simd256_impl&lt; true, true, true, 4 &gt;</a>	721
<a href="#">Simd256_impl&lt; true, true, true, 8 &gt;</a>	740
<a href="#">Simd256fp_base</a>	750
<a href="#">Simd256i_base</a>	751
<a href="#">Simd512_impl&lt; ArithType, Int, Signed, Size &gt;</a>	751
<a href="#">Simd512_impl&lt; true, false, true, 4 &gt;</a>	751
<a href="#">Simd512_impl&lt; true, false, true, 8 &gt;</a>	751
<a href="#">Simd512_impl&lt; true, true, false, 8 &gt;</a>	759
<a href="#">Simd512_impl&lt; true, true, true, 8 &gt;</a>	770
<a href="#">Simd512i_base</a>	781
<a href="#">SimdChooser&lt; T, bool, bool &gt;</a>	782
<a href="#">SimdChooser&lt; T, false, b &gt;</a>	782
<a href="#">SimdChooser&lt; T, true, false &gt;</a>	783
<a href="#">SimdChooser&lt; T, true, true &gt;</a>	783

<code>simdToType&lt; T &gt;</code>	783
<code>Single</code>	783
<code>Sparse&lt; Field, SparseMatrix_t, IdxT, PtrT &gt;</code>	783
<code>Sparse&lt; _Field, SparseMatrix_t::COO &gt;</code>	784
<code>Sparse&lt; _Field, SparseMatrix_t::COO_ZO &gt;</code>	785
<code>Sparse&lt; _Field, SparseMatrix_t::CSR &gt;</code>	787
<code>Sparse&lt; _Field, SparseMatrix_t::CSR_HYB &gt;</code>	789
<code>Sparse&lt; _Field, SparseMatrix_t::CSR_ZO &gt;</code>	790
<code>Sparse&lt; _Field, SparseMatrix_t::ELL &gt;</code>	792
<code>Sparse&lt; _Field, SparseMatrix_t::ELL_simd &gt;</code>	794
<code>Sparse&lt; _Field, SparseMatrix_t::ELL_simd_ZO &gt;</code>	795
<code>Sparse&lt; _Field, SparseMatrix_t::ELL_ZO &gt;</code>	797
<code>Sparse&lt; _Field, SparseMatrix_t::HYB_ZO &gt;</code>	799
<code>Sparse&lt; _Field, SparseMatrix_t::SELL &gt;</code>	800
<code>Sparse&lt; _Field, SparseMatrix_t::SELL_ZO &gt;</code>	802
<code>SpMat&lt; Field, flag &gt;</code>	804
<code>StatsMatrix</code>	805
<code>support_fast_mod&lt; T &gt;</code>	808
<code>support_fast_mod&lt; double &gt;</code>	808
<code>support_fast_mod&lt; float &gt;</code>	809
<code>support_fast_mod&lt; int64_t &gt;</code>	809
<code>support_simd&lt; T &gt;</code>	809
<code>support_simd_add&lt; T &gt;</code>	810
<code>support_simd_mod&lt; T &gt;</code>	810
<code>Test&lt; Elt &gt;</code>	810
<code>TestOneMethod&lt; Simd &gt;</code>	813
<code>tfn_minus</code>	816
<code>tfn_minus_eq</code>	816
<code>tfn_mul</code>	817
<code>tfn_mul_eq</code>	817
<code>tfn_plus</code>	817
<code>tfn_plus_eq</code>	818
<code>Threads</code>	818
<code>ThreeD</code>	818
<code>ThreeDAdaptive</code>	818
<code>ThreeDInPlace</code>	818
<code>TRSMHelper&lt; ReclterTrait, ParSeqTrait &gt;</code>	
<code>TRSM Helper</code>	819
<code>TwoD</code>	820
<code>TwoDAdaptive</code>	820
<code>UnparametricTag</code>	
If the field uses a representation with infix operators	820
<code>width&lt; T &gt;</code>	820
<code>width&lt; double &gt;</code>	821
<code>width&lt; float &gt;</code>	821
<code>Winograd</code>	821
<code>WinogradPar</code>	821



# Chapter 13

## File Index

### 13.1 File List

Here is a list of all files with brief descriptions:

101-fgemm.C	823
2x2-fgemm.C	823
2x2-ftsrv.C	824
2x2-pluq.C	824
align-allocator.h	825
args-parser.h	825
arithprog.C	827
benchmark-charpoly-mp.C	828
benchmark-charpoly.C	828
benchmark-checkers.C	829
benchmark-dgemm.C	830
benchmark-dgetrf.C	831
benchmark-dgetri.C	832
benchmark-dsytrf.C	833
benchmark-dtrsm.C	834
benchmark-dtrtri.C	835
benchmark-fadd-lvl2.C	835
benchmark-fdot.C	836
benchmark-fgemm-mp.C	837
benchmark-fgemm-rns.C	838
benchmark-fgemm.C	840
benchmark-fgemv-mp.C	840
benchmark-fgemv.C	841
benchmark-fgesv.C	844
benchmark-fsyr2k.C	845
benchmark-fsyrk.C	846
benchmark-fsytrf.C	846
benchmark-ftsrm-mp.C	847
benchmark-ftsrm.C	848
benchmark-ftsrv.C	848
benchmark-fttrtri.C	849
benchmark-inverse.C	850
benchmark-lqup-mp.C	850
benchmark-lqup.C	851
benchmark-pluq.C	852

benchmark-quasisep.C	853
benchmark-storage-transpose.C	854
benchmark-wino.C	855
bit_manipulation.h	855
blockcuts.inl	856
cast.h	858
cblas.C	858
autotune/charpoly.C	859
examples/charpoly.C	860
checker_charpoly.inl	860
checker_det.inl	861
checker_empty.h	861
checker_fgemm.inl	861
checker_ftrsm.inl	862
checker_invert.inl	862
checker_pluq.inl	863
checkers.doxy	863
checkers_fflas.h	863
checkers_fflas.inl	864
checkers_ffpack.h	864
checkers_ffpack.inl	865
clapack.C	865
config-blas.h	866
config.h	873
fflas-ffpack/config.h	877
coo.h	881
coo_spm্ম.inl	882
coo_spmmv.inl	883
coo_utils.inl	884
csr.h	884
csr_hyb.h	885
csr_hyb_pspmm.inl	885
csr_hyb_pspmv.inl	886
csr_hyb_spm্ম.inl	887
csr_hyb_spmmv.inl	887
csr_hyb_utils.inl	888
csr_pspmm.inl	888
csr_pspmv.inl	889
csr_spm্ম.inl	890
csr_spmmv.inl	891
csr_utils.inl	892
cuda.C	893
debug.h	
Various utilities for debugging	893
det.C	894
ell.h	894
ell_pspmm.inl	895
ell_pspmv.inl	896
ell_simd.h	897
ell_simd_pspmv.inl	897
ell_simd_spmmv.inl	898
ell_simd_utils.inl	899
ell_spm্ম.inl	899
ell_spmmv.inl	900
ell_utils.inl	901
fblas.C	902
fflas-101_1.C	903
fflas-101_3.C	903

fflas-ffpack-config.h	
Defaults for optimised values	904
fflas-ffpack-default-thresholds.h	904
fflas-ffpack-thresholds.h	905
fflas-ffpack.doxy	905
fflas-ffpack.h	
Includes <a href="#">FFLAS</a> and <a href="#">FFPACK</a>	905
fflas.doxy	906
fflas.h	
<b>Finite Field Linear Algebra Subroutines</b>	906
fflas_101.C	907
fflas_101_lvl1.C	907
fflas_bounds.inl	908
fflas_c.h	909
fflas_enum.h	922
fflas_fadd.h	922
fflas_fadd.inl	924
fflas_fassign.h	925
fflas_fassign.inl	925
fflas_faxpy.inl	926
fflas_fdot.inl	927
fflas_fgemm.inl	928
fflas_fgemv.inl	930
fflas_fgemv_mp.inl	932
fflas_fger.inl	932
fflas_fger_mp.inl	934
fflas_freduce.h	934
fflas_freduce.inl	935
fflas_freduce_mp.inl	937
fflas_freivalds.inl	937
fflas_fscal.h	938
fflas_fscal.inl	938
fflas_fscal_mp.inl	940
fflas_fsyr2k.inl	940
fflas_fsyrk.inl	941
fflas_fsyrk_strassen.inl	943
fflas_ftmm.inl	944
fflas_ftsm.inl	944
fflas_ftsm_mp.inl	
Triangular system with matrix right hand side over multiprecision domain (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ )	945
fflas_ftsv.inl	946
fflas_helpers.inl	946
fflas_intrinsic.h	948
fflas_io.h	948
fflas_L1_inst.C	948
fflas_L1_inst.h	950
fflas_L1_inst_implem.inl	951
fflas_L2_inst.C	952
fflas_L2_inst.h	953
fflas_L2_inst_implem.inl	955
fflas_L3_inst.C	956
fflas_L3_inst.h	957
fflas_L3_inst_implem.inl	959
fflas_level1.inl	960
fflas_level2.inl	962
fflas_level3.inl	965

<a href="#">fflas_lv1.C</a>	
C functions calls for level 1 <a href="#">FFLAS</a> in fflas-c.h	967
<a href="#">fflas_lv2.C</a>	
C functions calls for level 2 <a href="#">FFLAS</a> in fflas-c.h	972
<a href="#">fflas_lv3.C</a>	
C functions calls for level 3 <a href="#">FFLAS</a> in fflas-c.h	977
<a href="#">fflas_memory.h</a>	979
<a href="#">fflas_pfgemm.inl</a>	980
<a href="#">fflas_pftrsm.inl</a>	981
<a href="#">fflas_plevel1.h</a>	981
<a href="#">fflas_randommatrix.h</a>	982
<a href="#">fflas_simd.h</a>	984
<a href="#">fflas_sparse.C</a>	
C functions calls for level 1.5 and 2.5 <a href="#">FFLAS</a> in fflas-c.h	986
<a href="#">fflas_sparse.h</a>	986
<a href="#">fflas_sparse.inl</a>	991
<a href="#">fflas_transpose.h</a>	
Transpose the storage of the matrix (switch between row and col major mode)	993
<a href="#">ffpack-fgesv.C</a>	994
<a href="#">ffpack-solve.C</a>	994
<a href="#">ffpack.C</a>	
C functions calls for <a href="#">FFPACK</a> in ffpack-c.h	995
<a href="#">ffpack.doxy</a>	1016
<a href="#">ffpack.h</a>	
Set of elimination based routines for dense linear algebra	1016
<a href="#">ffpack.inl</a>	1026
<a href="#">ffpack_bruhatgen.inl</a>	1027
<a href="#">ffpack_c.h</a>	1028
<a href="#">ffpack_charpoly.inl</a>	1049
<a href="#">ffpack_charpoly_danilevski.inl</a>	1050
<a href="#">ffpack_charpoly_kgfast.inl</a>	1051
<a href="#">ffpack_charpoly_kgfastgeneralized.inl</a>	1051
<a href="#">ffpack_charpoly_kglu.inl</a>	1052
<a href="#">ffpack_charpoly_mp.inl</a>	1052
<a href="#">ffpack_det_mp.inl</a>	1053
<a href="#">ffpack_echelonforms.inl</a>	1054
<a href="#">ffpack_fgesv.inl</a>	1055
<a href="#">ffpack_fgets.inl</a>	1056
<a href="#">ffpack_frobenius.inl</a>	1056
<a href="#">ffpack_fsytrf.inl</a>	1057
<a href="#">ffpack_ftrssyr2k.inl</a>	1058
<a href="#">ffpack_ftrstr.inl</a>	1059
<a href="#">ffpack_fttr.inl</a>	1059
<a href="#">ffpack_inst.C</a>	1060
<a href="#">ffpack_inst.h</a>	1061
<a href="#">ffpack_inst_implem.inl</a>	1063
<a href="#">ffpack_invert.inl</a>	1066
<a href="#">ffpack_krylovelim.inl</a>	1066
<a href="#">ffpack_ludivine.inl</a>	1067
<a href="#">ffpack_ludivine_mp.inl</a>	1068
<a href="#">ffpack_minpoly.inl</a>	1068
<a href="#">ffpack_permutation.inl</a>	1069
<a href="#">ffpack_pluq.inl</a>	1071
<a href="#">ffpack_pluq_mp.inl</a>	1072
<a href="#">ffpack_ppluq.inl</a>	1073
<a href="#">ffpack_rankprofiles.inl</a>	1074
<a href="#">fgemm_classical.inl</a>	1075

fgemm_classical_mp.inl	
Matrix multiplication with multiprecision input (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ )	1075
fgemm_winograd.inl	1077
field-traits.h	
Field Traits	1079
field.doxy	1081
flimits.h	1081
fsyrk.C	1082
fsytrf.C	1083
fttrtri.C	1084
hyb_zo.h	1085
hyb_zo_pspmm.inl	1085
hyb_zo_pspmv.inl	1086
hyb_zo_spmmm.inl	1086
hyb_zo_spmv.inl	1087
hyb_zo_utils.inl	1088
igemm.doxy	1088
igemm.h	1088
igemm.inl	1089
igemm_kernels.h	1089
igemm_kernels.inl	1090
igemm_tools.h	1091
igemm_tools.inl	1091
interfaces.doxy	1092
kaapi_routines.inl	1092
lapack.C	1092
mainpage.doxy	1093
Matio.h	1093
matmul.C	1093
matmul.doxy	1094
parallel.h	1094
pfgemm_variants.inl	1101
pfgemv.inl	1102
autotune/pluq.C	1102
examples/pluq.C	1103
rank.C	1104
read_sparse.h	1104
regression-check.C	1105
rns-double-elt.h	
Rns elt structure with double support	1106
rns-double-recint.inl	1107
rns-double.h	
Rns structure with double support	1107
rns-double.inl	1108
rns-integer-mod.h	
Representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)	1108
rns-integer.h	
Representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)	1109
rns.h	1110
rns.inl	1110
schedule_bini.inl	
Bini implementation	1110
schedule_winograd.inl	1111
schedule_winograd_acc.inl	1112
schedule_winograd_acc_ip.inl	1112
schedule_winograd_ip.inl	1113
sell.h	1114
sell_pspmv.inl	1114

<a href="#">sell_spmv.inl</a>	1115
<a href="#">sell_utils.inl</a>	1116
<a href="#">simd.doxy</a>	1116
<a href="#">simd128.inl</a>	1116
<a href="#">simd128_double.inl</a>	1117
<a href="#">simd128_float.inl</a>	1118
<a href="#">simd128_int16.inl</a>	1118
<a href="#">simd128_int32.inl</a>	1118
<a href="#">simd128_int64.inl</a>	1119
<a href="#">simd256.inl</a>	1119
<a href="#">simd256_double.inl</a>	1120
<a href="#">simd256_float.inl</a>	1121
<a href="#">simd256_int16.inl</a>	1121
<a href="#">simd256_int32.inl</a>	1121
<a href="#">simd256_int64.inl</a>	1122
<a href="#">simd512.inl</a>	1123
<a href="#">simd512_double.inl</a>	1123
<a href="#">simd512_float.inl</a>	1124
<a href="#">simd512_int32.inl</a>	1124
<a href="#">simd512_int64.inl</a>	1125
<a href="#">simd_modular.inl</a>	1125
<a href="#">solve.C</a>	1125
<a href="#">sparse_matrix_traits.h</a>	1126
<a href="#">test-charpoly-check.C</a>	1127
<a href="#">test-charpoly.C</a>	1128
<a href="#">test-compressQ.C</a>	1129
<a href="#">test-det-check.C</a>	1130
<a href="#">test-det.C</a>	1131
<a href="#">test-echelon.C</a>	1131
<a href="#">test-fadd.C</a>	1134
<a href="#">test-fdot.C</a>	1135
<a href="#">test-fgemm-check.C</a>	1136
<a href="#">test-fgemm.C</a>	1138
<a href="#">test-fgemv.C</a>	1140
<a href="#">test-fger.C</a>	1142
<a href="#">test-fgesv.C</a>	1144
<a href="#">test-finit.C</a>	1145
<a href="#">test-fscal.C</a>	1146
<a href="#">test-fsyr2k.C</a>	1147
<a href="#">test-fsyrr.C</a>	1149
<a href="#">test-fsytrf.C</a>	1151
<a href="#">test-ftermm.C</a>	1152
<a href="#">test-ftermv.C</a>	1153
<a href="#">test-ftersm-check.C</a>	1155
<a href="#">test-ftersm.C</a>	1155
<a href="#">test-fterssyr2k.C</a>	1157
<a href="#">test-fterstr.C</a>	1158
<a href="#">test-ftersv.C</a>	1159
<a href="#">test-ftertri.C</a>	1160
<a href="#">test-interfaces-c.c</a>	1161
<a href="#">test-invert-check.C</a>	1162
<a href="#">test-io.C</a>	1162
<a href="#">test-lu.C</a>	1163
<a href="#">test-maxdelayeddim.C</a>	1168
<a href="#">test-minpoly.C</a>	1168
<a href="#">test-multifile1.C</a>	1169
<a href="#">test-multifile2.C</a>	1169
<a href="#">test-nullspace.C</a>	1170

test-permutations.C	1171
test-plug-check.C	1172
test-quasisep.C	1173
test-rankprofiles.C	1174
test-rpm.C	1175
test-simd.C	1176
test-solve.C	1180
test-storage-transpose.C	1180
test-utils.h	1181
timer.h	1182
utils.h	1182
winograd.C	1182



## Chapter 14

# Module Documentation

### 14.1 CHECKER

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

### 14.2 FFLAS

The C-style wrapper of BLAS for finite field linear algebra.

The C-style wrapper of BLAS for finite field linear algebra.

[FFLAS](#), Finite Field Linear Algebra Subroutines, provide basic linear algebra subroutines based on the BLAS interface. Therefore, the specifications are in C style; only the field given as a template parameter requires C++.

As much as possible, these routines use [ATLAS/BLAS](#) computations and achieve therefore high efficiency.

### 14.3 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

#### Files

- file [schedule\\_bini.inl](#)  
*Bini implementation.*

#### 14.3.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

[Todo](#) biblio

## 14.4 SIMD wrapper

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

**Todo** biblio

## 14.5 FFLAS-FFPACK

the [FFLAS FFPACK](#) library

### Modules

- [FFLAS](#)  
*The C-style wrapper of BLAS for finite field linear algebra.*
- [Interfaces](#)  
*Intefaces for FFLAS-FFPACK.*

### 14.5.1 Detailed Description

the [FFLAS FFPACK](#) library

C++ header library for fast exact dense linear algebra

See also

[FFLAS](#)  
[FFPACK](#)

## 14.6 FFPACK

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

## 14.7 FFLAS-FFPACK fields

fields in the FFLAS-FFPACK library

## Files

- file [rns-double-elt.h](#)  
*rns elt structure with double support*
- file [rns-double.h](#)  
*rns structure with double support*
- file [rns-integer-mod.h](#)  
*representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns-integer.h](#)  
*representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns.h](#)

### 14.7.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

[Todo](#) biblio

## 14.8 RNS

just include them all

just include them all

## 14.9 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

libs



## Chapter 15

# Namespace Documentation

### 15.1 FFLAS Namespace Reference

#### Namespaces

- [\\_ftranspose\\_impl](#)
- [BLAS3](#)
- [csr\\_hyb\\_details](#)
- [CuttingStrategy](#)
- [details](#)
- [details\\_spmv](#)
- [ElementCategories](#)
- [FieldCategories](#)

*Traits and categories will need to be placed in a proper file later.*

- [MMHelperAlgo](#)
- [ModeCategories](#)

*Specifies the mode of action for an algorithm w.r.t.*

- [ParSeqHelper](#)

*ParSeqHelper for both fgemm and ftrsm.*

- [Protected](#)
- [sell\\_details](#)
- [sparse\\_details](#)
- [sparse\\_details\\_impl](#)
- [StrategyParameter](#)
- [StructureHelper](#)

*StructureHelper for ftrsm.*

- [vectorised](#)

#### Data Structures

- struct [Checker\\_Empty](#)
- class [CheckerImplem\\_fgemm](#)
- class [CheckerImplem\\_ftrsm](#)
- struct [support\\_simd\\_add](#)
- struct [MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >](#)
- struct [MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)

- struct [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) >
- struct [support\\_simd\\_mod](#)
- struct [support\\_fast\\_mod](#)
- struct [support\\_fast\\_mod](#)< float >
- struct [support\\_fast\\_mod](#)< double >
- struct [support\\_fast\\_mod](#)< int64\_t >
- struct [AlgoChooser](#)
- struct [AlgoChooser](#)< [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) >
- struct [MMHelper](#)
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) >

*FGEMM Helper for Default and ConvertTo modes of operation.*

- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) >
- struct [TRSMHelper](#)

*TRSM Helper.*

- struct [support\\_simd](#)
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::COO](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::COO\\_ZO](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::ELL](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::ELL\\_ZO](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::ELL\\_simd](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::ELL\\_simd\\_ZO](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) >
- struct [readMyMachineType](#)
- struct [readMyMachineType](#)< [Field](#), [mpz\\_t](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::SELL](#) >
- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::SELL\\_ZO](#) >
- struct [isSparseMatrix](#)
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::COO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::COO\\_ZO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::SELL](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::SELL\\_ZO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_simd](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_simd\\_ZO](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > >
- struct [isSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > >
- struct [isZOSparseMatrix](#)
- struct [isZOSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > >
- struct [isZOSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::COO\\_ZO](#) > >
- struct [isZOSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#) > >
- struct [isZOSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::SELL\\_ZO](#) > >
- struct [isZOSparseMatrix](#)< [Field](#), [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_simd\\_ZO](#) > >
- struct [isSparseMatrixSimdFormat](#)
- struct [isSparseMatrixMKLFormat](#)
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)

- struct [tfn\\_plus\\_eq](#)
- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl](#)
- struct [has\\_mul\\_impl](#)
- struct [has\\_mul\\_eq\\_impl](#)
- struct [has\\_plus\\_eq\\_impl](#)
- struct [has\\_minus\\_eq\\_impl](#)
- struct [has\\_minus\\_impl](#)
- struct [has\\_operation](#)
- struct [StatsMatrix](#)
- struct [Sparse](#)
- struct [HelperFlag](#)
- struct [CsrMat](#)
- struct [CooMat](#)
- struct [EiIMat](#)
- struct [SpMat](#)
- struct [BlockTransposeSIMD](#)
- struct [ElementTraits](#)

*ElementTraits.*

- struct [ElementTraits](#)< float >
- struct [ElementTraits](#)< double >
- struct [ElementTraits](#)< int8\_t >
- struct [ElementTraits](#)< int16\_t >
- struct [ElementTraits](#)< int32\_t >
- struct [ElementTraits](#)< int64\_t >
- struct [ElementTraits](#)< uint8\_t >
- struct [ElementTraits](#)< uint16\_t >
- struct [ElementTraits](#)< uint32\_t >
- struct [ElementTraits](#)< uint64\_t >
- struct [ElementTraits](#)< Givaro::Integer >
- struct [ElementTraits](#)< Reclnt::rint< K > >
- struct [ElementTraits](#)< Reclnt::ruint< K > >
- struct [ElementTraits](#)< Reclnt::rmint< K, MG > >
- struct [ElementTraits](#)< FFPACK::rns\_double\_elt >
- struct [ModeTraits](#)

*ModeTraits.*

- struct [ModeTraits](#)< Givaro::Modular< Element, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< int64\_t, uint64\_t > >
- struct [ModeTraits](#)< Givaro::Modular< int8\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< int16\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< int32\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< uint8\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< uint16\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< uint32\_t, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< Givaro::Integer, Compute > >
- struct [ModeTraits](#)< Givaro::Modular< Reclnt::ruint< K >, Compute > >
- struct [ModeTraits](#)< Givaro::ModularBalanced< Element > >
- struct [ModeTraits](#)< Givaro::ModularBalanced< int8\_t > >
- struct [ModeTraits](#)< Givaro::ModularBalanced< int16\_t > >
- struct [ModeTraits](#)< Givaro::ModularBalanced< int32\_t > >
- struct [ModeTraits](#)< Givaro::ModularBalanced< Givaro::Integer > >
- struct [ModeTraits](#)< Givaro::ZRing< Givaro::Integer > >
- struct [ModeTraits](#)< Givaro::ZRing< float > >
- struct [ModeTraits](#)< Givaro::ZRing< double > >

- struct [ModeTraits](#)< Givaro::Montgomery< T > >
- struct [FieldTraits](#)  
    *FieldTrait.*
- struct [FieldTraits](#)< Givaro::ZRing< Reclnt::ruint< K > > >
- struct [FieldTraits](#)< Givaro::Modular< Element > >
- struct [FieldTraits](#)< Givaro::ModularBalanced< Element > >
- struct [FieldTraits](#)< Givaro::ZRing< double > >
- struct [FieldTraits](#)< Givaro::ZRing< float > >
- struct [FieldTraits](#)< Givaro::ZRing< int16\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< uint16\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< int32\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< uint32\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< int64\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< uint64\_t > >
- struct [FieldTraits](#)< Givaro::ZRing< Givaro::Integer > >
- struct [FieldTraits](#)< FFPACK::RNSInteger< T > >
- struct [FieldTraits](#)< FFPACK::RNSIntegerMod< T > >
- struct [associatedDelayedField](#)
- struct [associatedDelayedField](#)< const Givaro::Modular< T, X > >
- struct [associatedDelayedField](#)< const Givaro::ModularBalanced< T > >
- struct [associatedDelayedField](#)< const Givaro::ZRing< T > >
- struct [associatedDelayedField](#)< const FFPACK::RNSIntegerMod< RNS > >
- struct [ForStrategy1D](#)
- struct [ForStrategy2D](#)

## Typedefs

- template<class Field >  
    using [Checker\\_fgemm](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
    using [Checker\\_ftsm](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
    using [ForceCheck\\_fgemm](#) = CheckerImplem\_fgemm< Field >
- template<class Field >  
    using [ForceCheck\\_ftsm](#) = CheckerImplem\_ftsm< Field >
- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
    using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_impl](#)< T > >::type
- template<class T >  
    using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_minus\\_impl](#)< T > >::type
- template<class T >  
    using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copyable\_↵\_assignable< T > >::type
- template<class T >  
    using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_eq\\_impl](#)< T > >::type

- `template<class T >`  
`using has_minus_eq = typename std::conditional< std::is_arithmetic< T >::value, std::true_type,`  
`has_minus_eq_impl< T > >::type`
- `template<class T >`  
`using has_mul = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_impl<`  
`T > >::type`
- `template<class T >`  
`using has_mul_eq = typename std::conditional< std::is_arithmetic< T >::value, std::true_type,`  
`has_mul_eq_impl< T > >::type`
- `typedef Givaro::Timer Timer`
- `typedef Givaro::BaseTimer BaseTimer`
- `typedef Givaro::UserTimer UserTimer`
- `typedef Givaro::SysTimer SysTimer`

## Enumerations

- `enum FFLAS_ORDER { FflasRowMajor =101 , FflasColMajor =102 }`  
*Storage by row or col ?*
- `enum FFLAS_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }`  
*Is matrix transposed ?*
- `enum FFLAS_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasLeftTri = 123 , FflasRightTri = 124 }`  
*Is triangular matrix's shape upper ?*
- `enum FFLAS_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }`  
*Is the triangular matrix implicitly unit diagonal ?*
- `enum FFLAS_SIDE { FflasLeft = 141 , FflasRight = 142 }`  
*On what side ?*
- `enum FFLAS_BASE { FflasDouble = 151 , FflasFloat = 152 , FflasGeneric = 153 }`  
*FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*
- `enum number_kind { zero =0 , one =1 , mone =-1 , other =2 }`
- `enum class SparseMatrix_t {`  
`CSR , CSR_ZO , CSC , CSC_ZO ,`  
`COO , COO_ZO , ELL , ELL_ZO ,`  
`SELL , SELL_ZO , ELL_simd , ELL_simd_ZO ,`  
`CSR_HYB , HYB_ZO }`
- `enum FFLAS_FORMAT {`  
`FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,`  
`FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }`

## Functions

- `Givaro::Integer InfNorm (const size_t M, const size_t N, const Givaro::Integer *A, const size_t Ida)`
- `template<class T >`  
`const T & min3 (const T &m, const T &n, const T &k)`
- `template<class T >`  
`const T & max3 (const T &m, const T &n, const T &k)`
- `template<class T >`  
`const T & min4 (const T &m, const T &n, const T &k, const T &l)`
- `template<class T >`  
`const T & max4 (const T &m, const T &n, const T &k, const T &l)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`

- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadding for symmetric matrices*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*
- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  
*fassign :  $x \leftarrow y$ .*

- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$
- `template<class Field >`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::ConstElement_ptr a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t idx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)`
- `template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field, class T >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fdot: \text{dot product } x^T y.$$

- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`
- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >`  
`Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >`  
`RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >`  
`RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`
- `Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`

- `template<class ParSeq >`  
`Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<size_t K1, size_t K2, class ParSeq >`  
`RecInt::ruint< K1 > * fgemm (const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const RecInt::ruint< K1 > alpha, const RecInt::ruint< K1 > *A, const size_t lda, const RecInt::ruint< K1 > *B, const size_t ldb, RecInt::ruint< K1 > beta, RecInt::ruint< K1 > *C, const size_t ldc, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<class Field, class ModeT >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)`
- `template<class Field, class ModeT, class Cut, class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)`
- `template<typename Field, class Cut, class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fgemm: Field GENERAL Matrix Multiply.*
- `template<typename Field, class ModeT, class ParSeq >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`

- tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [ModeCategories::DelayedTag](#), [ParSeqHelper::Sequential](#) > &H)
- `template<class Field >`  
[Field::Element\\_ptr](#) [fsquare](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsquare: Squares a matrix.*
  - `template<> double * fsquare` (const [Givaro::ModularBalanced](#)< double > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double beta, double \*C, const size\_t ldc)
  - `template<> float * fsquare` (const [Givaro::ModularBalanced](#)< float > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t n, const float alpha, const float \*A, const size\_t lda, const float beta, float \*C, const size\_t ldc)
  - `template<> double * fsquare` (const [Givaro::Modular](#)< double > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double beta, double \*C, const size\_t ldc)
  - `template<> float * fsquare` (const [Givaro::Modular](#)< float > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t n, const float alpha, const float \*A, const size\_t lda, const float beta, float \*C, const size\_t ldc)
  - `template<class Field >`  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::MachineFloatTag](#) > > &H)
  - `template<class Field >`  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DelayedTag](#) > &H)
  - `template<class Field >`  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
  - `template<class Field >`  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::LazyTag](#) > &H)
  - `template<class Field >`  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) TransA, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  
*finite prime Field GEneral Matrix Vector multiplication.*
  - `Givaro::ZRing< int64_t >::Element_ptr fgemv` (const [Givaro::ZRing](#)< int64\_t > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*X, const size\_t incX, const int64\_t beta, const int64\_t \*Y, const size\_t incY, [MMHelper](#)< [Givaro::ZRing](#)< int64\_t >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
  - `Givaro::DoubleDomain::Element_ptr fgemv` (const [Givaro::DoubleDomain](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [Givaro::DoubleDomain::Element](#) alpha, const [Givaro::DoubleDomain::ConstElement\\_ptr](#) A, const size\_t lda, const [Givaro::DoubleDomain::ConstElement\\_ptr](#) X, const size\_t incX, const [Givaro::DoubleDomain::Element](#) beta, [Givaro::DoubleDomain::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Givaro::DoubleDomain](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)

- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::ConstElement_ptr` X, const `size_t` incX, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultBoundedTag` > &H)
- `Givaro::FloatDomain::Element_ptr fgemv` (const `Givaro::FloatDomain` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `Givaro::FloatDomain::Element` alpha, const `Givaro::FloatDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::FloatDomain::ConstElement_ptr` X, const `size_t` incX, const `Givaro::FloatDomain::Element` beta, `Givaro::FloatDomain::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Givaro::FloatDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field, class Cut, class Param >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::ConstElement_ptr` X, const `size_t` incX, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` Y, const `size_t` incY, `ParSeqHelper::Parallel`< `Cut`, `Param` > &parH)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::ConstElement_ptr` X, const `size_t` incX, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` Y, const `size_t` incY, `ParSeqHelper::Sequential` &seqH)
- `FFPACK::rns_double::Element_ptr fgemv` (const `FFPACK::RNSInteger`< `FFPACK::rns_double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `FFPACK::rns_double::Element` alpha, `FFPACK::rns_double::ConstElement_ptr` A, const `size_t` lda, `FFPACK::rns_double::ConstElement_ptr` X, const `size_t` incX, const `FFPACK::rns_double::Element` beta, `FFPACK::rns_double::Element_ptr` Y, const `size_t` incY, `MMHelper`< `FFPACK::RNSInteger`< `FFPACK::rns_double` >, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `FFPACK::rns_double::Element_ptr fgemv` (const `FFPACK::RNSIntegerMod`< `FFPACK::rns_double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `FFPACK::rns_double::Element` alpha, `FFPACK::rns_double::ConstElement_ptr` A, const `size_t` lda, `FFPACK::rns_double::ConstElement_ptr` X, const `size_t` incX, const `FFPACK::rns_double::Element` beta, `FFPACK::rns_double::Element_ptr` Y, const `size_t` incY, `MMHelper`< `FFPACK::RNSIntegerMod`< `FFPACK::rns_double` >, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `Givaro::Integer * fgemv` (const `Givaro::ZRing`< `Givaro::Integer` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const `Givaro::Integer` alpha, `Givaro::Integer *A`, const `size_t` lda, `Givaro::Integer *X`, const `size_t` ldx, `Givaro::Integer` beta, `Givaro::Integer *Y`, const `size_t` ldy, `MMHelper`< `Givaro::ZRing`< `Givaro::Integer` >, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::RNSElementTag` > > &H)
- `Givaro::Integer * fgemv` (const `Givaro::Modular`< `Givaro::Integer` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const `Givaro::Integer` alpha, `Givaro::Integer *A`, const `size_t` lda, `Givaro::Integer *X`, const `size_t` ldx, `Givaro::Integer` beta, `Givaro::Integer *Y`, const `size_t` ldy, `MMHelper`< `Givaro::Modular`< `Givaro::Integer` >, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::RNSElementTag` > > &H)
- `template<size_t K1, size_t K2, class ParSeq >`  
`Reclnt::ruint`< `K1` > \* `fgemv` (const `Givaro::Modular`< `Reclnt::ruint`< `K1` >, `Reclnt::ruint`< `K2` > > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const `Reclnt::ruint`< `K1` > alpha, const `Reclnt::ruint`< `K1` > \*A, const `size_t` lda, const `Reclnt::ruint`< `K1` > \*X, const `size_t` incx, `Reclnt::ruint`< `K1` > beta, `Reclnt::ruint`< `K1` > \*Y, const `size_t` incy, `MMHelper`< `Givaro::Modular`< `Reclnt::ruint`< `K1` >, `Reclnt::ruint`< `K2` > >, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::RNSElementTag` >, `ParSeq` > &H)
- `template<class Field >`  
`void fger` (const `Field` &F, const `size_t` M, const `size_t` N, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` x, const `size_t` incx, `typename` `Field::ConstElement_ptr` y, const `size_t` incy, `typename` `Field::Element_ptr` A, const `size_t` lda)  
*fger: rank one update of a general matrix*
- `template<class Field >`  
`void fger` (const `Field` &F, const `size_t` M, const `size_t` N, const `typename` `Field::Element` alpha, type-

- name `Field::ConstElement_ptr` x, const size\_t incx, typename `Field::ConstElement_ptr` y, const size\_t incy, typename `Field::Element_ptr` A, const size\_t lda, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::MachineFloatTag` > > &H)
- `template<class Field, class AnyTag >`  
`void fger` (const `Field` &F, const size\_t M, const size\_t N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` x, const size\_t incx, typename `Field::ConstElement_ptr` y, const size\_t incy, typename `Field::Element_ptr` A, const size\_t lda, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `AnyTag` > &H)
  - `void fger` (const `Givaro::DoubleDomain` &F, const size\_t M, const size\_t N, const `Givaro::DoubleDomain::Element` alpha, const `Givaro::DoubleDomain::ConstElement_ptr` x, const size\_t incx, const `Givaro::DoubleDomain::ConstElement_ptr` y, const size\_t incy, `Givaro::DoubleDomain::Element_ptr` A, const size\_t lda, `MMHelper`< `Givaro::DoubleDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
  - `template<class Field >`  
`void fger` (const `Field` &F, const size\_t M, const size\_t N, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` x, const size\_t incx, const typename `Field::ConstElement_ptr` y, const size\_t incy, typename `Field::Element_ptr` A, const size\_t lda, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultBoundedTag` > &H)
  - `void fger` (const `Givaro::FloatDomain` &F, const size\_t M, const size\_t N, const `Givaro::FloatDomain::Element` alpha, const `Givaro::FloatDomain::ConstElement_ptr` x, const size\_t incx, const `Givaro::FloatDomain::ConstElement_ptr` y, const size\_t incy, `Givaro::FloatDomain::Element_ptr` A, const size\_t lda, `MMHelper`< `Givaro::FloatDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
  - `template<class Field >`  
`void fger` (const `Field` &F, const size\_t M, const size\_t N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` x, const size\_t incx, typename `Field::ConstElement_ptr` y, const size\_t incy, typename `Field::Element_ptr` A, const size\_t lda, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::LazyTag` > &H)
  - `template<class Field >`  
`void fger` (const `Field` &F, const size\_t M, const size\_t N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` x, const size\_t incx, typename `Field::ConstElement_ptr` y, const size\_t incy, typename `Field::Element_ptr` A, const size\_t lda, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DelayedTag` > &H)
  - `void fger` (const `Givaro::Modular`< `Givaro::Integer` > &F, const size\_t M, const size\_t N, const typename `Givaro::Integer` alpha, typename `Givaro::Integer` \*x, const size\_t incx, typename `Givaro::Integer` \*y, const size\_t incy, typename `Givaro::Integer` \*A, const size\_t lda, `MMHelper`< `Givaro::Modular`< `Givaro::Integer` >, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::RNSElementTag` > > &H)
  - `template<typename RNS >`  
`void fger` (const `FFPACK::RNSInteger`< `RNS` > &F, const size\_t M, const size\_t N, const typename `FFPACK::RNSInteger`< `RNS` >::Element alpha, typename `FFPACK::RNSInteger`< `RNS` >::Element\_ptr x, const size\_t incx, typename `FFPACK::RNSInteger`< `RNS` >::Element\_ptr y, const size\_t incy, typename `FFPACK::RNSInteger`< `RNS` >::Element\_ptr A, const size\_t lda, `MMHelper`< `FFPACK::RNSInteger`< `RNS` >, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
  - `template<typename RNS >`  
`void fger` (const `FFPACK::RNSIntegerMod`< `RNS` > &F, const size\_t M, const size\_t N, const typename `FFPACK::RNSIntegerMod`< `RNS` >::Element alpha, typename `FFPACK::RNSIntegerMod`< `RNS` >::Element\_ptr x, const size\_t incx, typename `FFPACK::RNSIntegerMod`< `RNS` >::Element\_ptr y, const size\_t incy, typename `FFPACK::RNSIntegerMod`< `RNS` >::Element\_ptr A, const size\_t lda, `MMHelper`< `FFPACK::RNSIntegerMod`< `RNS` >, `MMHelperAlgo::Classic` > &H)
  - `template<class Field >`  
`void freduce` (const `Field` &F, const size\_t n, typename `Field::ConstElement_ptr` Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)  

$$freduce\ x \leftarrow ymodF.$$
  - `template<class Field >`  
`void freduce` (const `Field` &F, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  

$$freduce\ x \leftarrow xmodF.$$
  - `template<class Field >`  
`void freduce_constoverride` (const `Field` &F, const size\_t m, typename `Field::ConstElement_ptr` A, const size\_t incX)

- `template<class Field , class ConstOtherElement_ptr >`  
`void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`
- `template<class Field >`  
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  
*fini* Initializes  $X$  in  $F^S$ .
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*freduce*  $A \leftarrow A \bmod F$ .
- `template<class Field >`  
`void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*freduce* for square symmetric matrices
- `template<class Field >`  
`void pfreduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const size_t numths)`
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*freduce*  $A \leftarrow B \bmod F$ .
- `template<class Field >`  
`void freduce_constoverride (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)`
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*fini*  $A \leftarrow B \bmod F$ .
- `template<class Field >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*fini* Initializes  $A$  in  $F^S$ .
- `template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, size_t inc)`
- `template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, FFPACK::rns_double::Element_ptr A, size_t lda)`
- `template<class Field >`  
`bool freivalds (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::ConstElement_ptr C, const size_t ldc)`  
*freivalds*: **Fre**ivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.
- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)`  
*fscal*  $x \leftarrow \alpha \cdot x$ .
- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  
*fscal*  $y \leftarrow \alpha \cdot x$ .
- `template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`

- `template<> void fscaln` (const Givaro::DoubleDomain &, const size\_t N, const Givaro::DoubleDomain::↵ Element a, [Givaro::DoubleDomain::Element\\_ptr](#) y, const size\_t incy)
- `template<> void fscaln` (const Givaro::FloatDomain &, const size\_t N, const Givaro::FloatDomain::Element a, [Givaro::FloatDomain::Element\\_ptr](#) y, const size\_t incy)
- `template<class Field >`  
void `fscaln` (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$fscaln\ A \leftarrow a \cdot A.$$
- `template<class Field >`  
void `fscal` (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
$$fscal\ B \leftarrow a \cdot A.$$
- `template<> void fscaln` (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t inc)
- `template<> void fscal` (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- `template<> void fscaln` (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- `template<> void fscal` (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t↵\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)
- `template<> void fscaln` (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t↵\_t n, const typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, const size\_t inc)
- `template<> void fscal` (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- `template<> void fscaln` (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- `template<> void fscal` (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t↵\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)
- `template<class Field >`  
[Field::Element\\_ptr](#) `fsyr2k` (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
$$fsyr2k: \text{Symmetric Rank } 2K \text{ update}$$
- `template<class Field >`  
[Field::Element\\_ptr](#) `fsyrk` (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
$$fsyrk: \text{Symmetric Rank } K \text{ update}$$
- `template<class Field >`  
[Field::Element\\_ptr](#) `fsyrk` (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const [ParSeqHelper::Sequential](#) seq)
- `template<class Field >`  
[Field::Element\\_ptr](#) `fsyrk` (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)

- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::MachineFloatTag` >, `ParSeqHelper::Sequential` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DelayedTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::LazyTag` > &H)
- `template<class Field , typename Mode >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::DivideAndConquer`, `Mode` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultBoundedTag` > &H)
- `Givaro::FloatDomain::Element_ptr fsyrk` (const `Givaro::FloatDomain` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `Givaro::FloatDomain::Element` alpha, `Givaro::FloatDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::FloatDomain::Element` beta, `Givaro::FloatDomain::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Givaro::FloatDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `Givaro::DoubleDomain::Element_ptr fsyrk` (const `Givaro::DoubleDomain` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `Givaro::DoubleDomain::Element` alpha, `Givaro::DoubleDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::DoubleDomain::Element` beta, `Givaro::DoubleDomain::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Givaro::DoubleDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` D, const `size_t` incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `size_t` threshold=`__FFLASFFPACK_FSYRK_THRESHOLD`)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` D, const `size_t` incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq, const `size_t` threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` D, const `size_t` incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Parallel`< `Cut`, `Param` > par, const `size_t` threshold)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const

size\_t lda, typename [Field::ConstElement\\_ptr](#) D, const size\_t incD, const std::vector< bool > &two←  
Block, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const size\_t  
threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)

*fsyrk: Symmetric Rank K update with diagonal scaling*

- template<class Field , class FieldTrait >  
void [computeS1S2](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) trans, const size\_t N, const size\_t K, const  
typename [Field::Element](#) x, const typename [Field::Element](#) y, typename [Field::Element\\_ptr](#) A, const size\_t  
\_t lda, typename [Field::Element\\_ptr](#) S, const size\_t lds, typename [Field::Element\\_ptr](#) T, const size\_t ldt,  
[MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldTrait](#) > &WH)
- template<class Field >  
[Field::Element\\_ptr](#) [fsyrk](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const  
size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const  
size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)<  
[Field](#), [MMHelperAlgo::Winograd](#), [ModeCategories::DelayedTag](#), [ParSeqHelper::Sequential](#) > &H)
- template<class Field , class Mode >  
[Field::Element\\_ptr](#) [fsyrk](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) UpLo, const [FFLAS\\_TRANSPOSE](#) trans, const  
size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const  
size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)<  
[Field](#), [MMHelperAlgo::Winograd](#), [Mode](#) > &H)
- template<class Field , class FieldTrait >  
[Field::Element\\_ptr](#) [fsyrk\\_strassen](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) uplo, const [FFLAS\\_TRANSPOSE](#)  
trans, const size\_t N, const size\_t K, const typename [Field::Element](#) y1, const typename [Field::Element](#)  
y2, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, const  
typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#),  
[MMHelperAlgo::Winograd](#), [FieldTrait](#) > &WH)
- template<class Field >  
void [ftrmm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha,  
typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
  
*ftrmm: **TRI**angular **M**atrix **M**ultiply.*
- template<class Field >  
void [ftrmm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha,  
typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t  
ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
  
*ftrmm: **TRI**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha \text{Bop}(A) + \text{beta}C$ .*
- template<class Field >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha,  
typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)
- template<class Field >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha,  
typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const  
[ParSeqHelper::Sequential](#) &PSH)
- template<class Field , class Cut , class Param >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha,  
typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const  
[ParSeqHelper::Parallel](#)< Cut, Param > &PSH)
- template<class Field , class ParSeqTrait = [ParSeqHelper::Sequential](#)>  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#)  
TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, type-  
name [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, [TRSMHelper](#)<  
[StructureHelper::Recursive](#), [ParSeqTrait](#) > &H)
- void [ftrsm](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#)  
Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const  
[Givaro::Integer](#) alpha, const [Givaro::Integer](#) \*A, const size\_t lda, [Givaro::Integer](#) \*B, const size\_t ldb)

- void `cblas_imptrsm` (const enum `FFLAS_ORDER` Order, const enum `FFLAS_SIDE` Side, const enum `FFLAS_UPLO` Uplo, const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_DIAG` Diag, const int M, const int N, const `FFPACK::rns_double_elt` alpha, `FFPACK::rns_double_elt_cstptr` A, const int lda, `FFPACK::rns_double_elt_ptr` B, const int ldb)
- template<class Field >  
void `ftrsv` (const Field &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::Element_ptr` X, int incX)  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*
- void `igemm` (const enum `FFLAS_ORDER` Order, const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_TRANSPOSE` TransB, const size\_t M, const size\_t N, const size\_t K, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*B, const size\_t ldb, const int64\_t beta, int64\_t \*C, const size\_t ldc)
- template<class Field, class OtherElement\_ptr >  
void `finit` (const Field &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)  
*finit  $x \leftarrow y \bmod F$ .*
- template<class Field, class OtherElement\_ptr >  
void `fconvert` (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename `Field::ConstElement_ptr` Y, const size\_t incY)  
*fconvert  $x \leftarrow y \bmod F$ .*
- template<class Field >  
void `fnegin` (const Field &F, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  
*fnegin  $x \leftarrow -x$ .*
- template<class Field >  
void `fneg` (const Field &F, const size\_t n, typename `Field::ConstElement_ptr` Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)  
*fneg  $x \leftarrow -y$ .*
- template<class Field >  
void `fzero` (const Field &F, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  
*fzero :  $A \leftarrow 0$ .*
- template<class Field, class RandIter >  
void `frand` (const Field &F, RandIter &G, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  
*frand :  $A \leftarrow \text{random}$ .*
- template<class Field >  
bool `fiszero` (const Field &F, const size\_t n, typename `Field::ConstElement_ptr` X, const size\_t incX)  
*fiszero : test  $X = 0$ .*
- template<class Field >  
bool `fequal` (const Field &F, const size\_t n, typename `Field::ConstElement_ptr` X, const size\_t incX, typename `Field::ConstElement_ptr` Y, const size\_t incY)  
*fequal : test  $X = Y$ .*
- template<class Field >  
void `faxpby` (const Field &F, const size\_t N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` X, const size\_t incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const size\_t incY)  
*faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .*
- template<typename Field, class Cut, class Param >  
`Field::Element` `fdot` (const Field &F, const size\_t N, typename `Field::ConstElement_ptr` X, const size\_t incX, typename `Field::ConstElement_ptr` Y, const size\_t incY, const `ParSeqHelper::Parallel`< Cut, Param > par)
- template<class Field >  
void `fswap` (const Field &F, const size\_t N, typename `Field::Element_ptr` X, const size\_t incX, typename `Field::Element_ptr` Y, const size\_t incY)  
*fswap:  $X \leftrightarrow Y$ .*
- template<class Field >  
void `fzero` (const Field &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)  
*fzero :  $A \leftarrow 0$ .*

- `template<class Field >`  
`void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*fzero :  $A \leftarrow 0$  for a triangular matrix.*
- `template<class Field , class Randlter >`  
`void frand (const Field &F, Randlter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*frand :  $A \leftarrow random$ .*
- `template<class Field >`  
`bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  
*fequal : test  $A = B$ .*
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)`  
*fiszero : test  $A = 0$ .*
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const typename Field::Element &d)`  
*creates a diagonal matrix*
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*creates a diagonal matrix*
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*finit Initializes  $A$  in  $F^{\$}$ .*
- `template<class Field , class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  
*fconvert  $A \leftarrow B \bmod F$ .*
- `template<class Field >`  
`void fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*fnegin  $A \leftarrow -A$ .*
- `template<class Field >`  
`void fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*fneg  $A \leftarrow -B$ .*
- `template<class Field >`  
`void faxpby (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t ldy)`  
*faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .*
- `template<class Field >`  
`void fmove (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .*
- `template<class Field >`  
`size_t bitsize (const Field &F, size_t M, size_t N, const typename Field::ConstElement_ptr A, size_t lda)`  
*bitsize: Computes the largest bitsize of the matrix' coefficients.*
- `template<> size_t bitsize< Givaro::ZRing< Givaro::Integer > > (const Givaro::ZRing< Givaro::Integer > &F, size_t M, size_t N, const Givaro::Integer *A, size_t lda)`
- `template<class Field >`  
`void ftrmv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)`

*ftsm*: *TRiangular Matrix Vector prodcut Computes*  $X \leftarrow \text{op}(A)X$

- template<class Field >  
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)  
  
*ftsm*: **TRiangular System solve with Matrix.**
- template<class Field , typename FieldTrait >  
Field::Element\_ptr **fsyrk\_strassen** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element y1, const typename Field::Element y2, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &H)
- template<typename Field >  
Field::Element\_ptr **pfgemm** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t numthreads=0)
- template<class Field >  
Field::Element \* **pfgemm\_1D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
Field::Element \* **pfgemm\_2D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
Field::Element \* **pfgemm\_3D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil, size\_t \*x)
- template<class Field >  
Field::Element\_ptr **pfgemm\_3D\_rec2** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil, size\_t \*x)
- template<class Field , class ModeTrait , class Strat , class Param >  
std::enable\_if<!std::is\_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > >::value, typename Field::Element\_ptr >::type **fgemm** (const Field &F, const FFLAS::FFLAS\_TRANSPOSE ta, const FFLAS::FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > > &H)
- template<class Field , class Cut , class Param >  
Field::Element\_ptr **ftsm** (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_TRANSPOSE TA, const FFLAS::FFLAS\_DIAG Diag, const size\_t m, const size\_t n, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class Field , class Cut , class Param >  
Field::Element\_ptr **ftsm** (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_TRANSPOSE TA, const FFLAS::FFLAS\_DIAG Diag, const size\_t m, const size\_t n, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)

- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::COO > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::CSR > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::CSR\_ZO > &A)`
- `template<class Field >`  
`std::ostream & sparse\_print (std::ostream &os, const Sparse< Field, SparseMatrix\_t::CSR > &A)`
- `template<class IndexT >`  
`void sparse\_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT >`  
`void sparse\_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix\_t::CSR\_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT , size_t RECINT_SIZE>`  
`void sparse\_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix\_t::CSR\_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT , size_t RECINT_SIZE>`  
`void sparse\_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::CSR\_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::CSR\_HYB > &A)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::CSR\_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::ELL > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::ELL\_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::ELL\_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::ELL\_simd > &A)`

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<typename _Field >`  
`std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field , bool sorted = true, bool read_integer = false>`  
`void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class Field >`  
`void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_floating_point< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()`
- `template<class T >`  
`int getDataType ()`
- `template<class Field >`  
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)`
- `template<class Field >`  
`void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)`
- `template<class Field >`  
`void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t ldA)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

- template<class It >  
double [computeDeviation](#) (It begin, It end)
- template<class Field >  
[StatsMatrix](#) [getStat](#) (const [Field](#) &F, const [index\\_t](#) \*row, const [index\\_t](#) \*col, typename [Field::ConstElement\\_ptr](#) val, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class Field , class SM >  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, const typename [Field::Element](#) &beta, typename [Field::Element\\_ptr](#) y)
- template<class Field , class SM >  
void [fspmm](#) (const [Field](#) &F, const SM &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, const typename [Field::Element](#) &beta, typename [Field::Element\\_ptr](#) y, int ldy)
- template<class Field , class enable = void>  
[Field::Residu\\_t](#) [maxCardinality](#) ()
- template<> [uint64\\_t](#) [maxCardinality](#)< [Givaro::Modular](#)< [int64\\_t](#) > > ()
- template<> [uint32\\_t](#) [maxCardinality](#)< [Givaro::Modular](#)< [int32\\_t](#) > > ()
- template<class Field >  
[Field::Residu\\_t](#) [minCardinality](#) ()
- template<> void [fflas\\_delete](#) ([FFPACK::rns\\_double\\_elt\\_ptr](#) A)
- template<> void [fflas\\_delete](#) ([FFPACK::rns\\_double\\_elt\\_cstptr](#) A)
- template<> [FFPACK::rns\\_double\\_elt\\_ptr](#) [fflas\\_new](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const [size\\_t](#) m, const [Alignment](#) align)
- template<> [FFPACK::rns\\_double\\_elt\\_ptr](#) [fflas\\_new](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, const [Alignment](#) align)
- template<typename RNS >  
void [finit\\_rns](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [size\\_t](#) k, const [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [RNS::Element\\_ptr](#) A)
- template<typename RNS >  
void [finit\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [size\\_t](#) k, const [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [RNS::Element\\_ptr](#) A)
- template<typename RNS >  
void [fconvert\\_rns](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [Givaro::Integer](#) alpha, [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [RNS::ConstElement\\_ptr](#) A)
- template<typename RNS >  
void [fconvert\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [Givaro::Integer](#) alpha, [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [RNS::ConstElement\\_ptr](#) A)
- template<> [FFPACK::rns\\_double\\_elt\\_ptr](#) [fflas\\_new](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const [size\\_t](#) m, const [Alignment](#) align)
- template<> [FFPACK::rns\\_double\\_elt\\_ptr](#) [fflas\\_new](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, const [Alignment](#) align)
- template<typename RNS >  
void [finit\\_rns](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [size\\_t](#) k, const [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr A)
- template<typename RNS >  
void [fconvert\\_rns](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const [size\\_t](#) m, const [size\\_t](#) n, [Givaro::Integer](#) alpha, [Givaro::Integer](#) \*B, const [size\\_t](#) ldb, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr A)
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) n, const [FFLAS\\_ELT](#) \*Y, const [size\\_t](#) incY, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) incX)  
$$\text{freduce } x \leftarrow x \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) n, const [FFLAS\\_ELT](#) \*Y, const [size\\_t](#) incY, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) incX)  
$$\text{freduce } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) n, const [FFLAS\\_ELT](#) \*Y, const [size\\_t](#) incY, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) incX)  
$$\text{finit } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [fconvert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) n, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) incX, const [FFLAS\\_ELT](#) \*Y, const [size\\_t](#) incY)

- $fconvert\ x \leftarrow y \bmod F.$ 
  - template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)
    - $fnegin\ x \leftarrow -x.$
  - template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)
    - $fneg\ x \leftarrow -y.$
  - template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)
    - $fzero : A \leftarrow 0.$
  - template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX)
    - $fiszero : test\ X = 0.$
  - template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)
    - $fequal : test\ X = Y.$
  - template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)
    - $fassign : x \leftarrow y.$
  - template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)
    - $fscaln\ x \leftarrow \alpha \cdot x.$
  - template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
    - $fscal\ y \leftarrow \alpha \cdot x.$
  - template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
    - $faxpy : y \leftarrow \alpha \cdot x + y.$
  - template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)
    - $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
  - template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
    - $fswap : X \leftrightarrow Y.$
  - template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
    -
  - template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
    -
  - template `INST_OR_DECL` void `faddn` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
    -
  - template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
    -
  - template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)
    - $fassign : A \leftarrow B.$
  - template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)
    - $fzero : A \leftarrow 0.$
  - template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb)
    - $fequal : test\ A = B.$

- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{fiszero} : \text{test } A = 0.$$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` &d)  

$$\text{creates a diagonal matrix}$$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{creates a diagonal matrix}$$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{freduce } A \leftarrow A \bmod F.$$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{freduce } A \leftarrow B \bmod F.$$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{finit } A \leftarrow B \bmod F.$$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{fnegin } A \leftarrow -A.$$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{fneg } A \leftarrow -B.$$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*A, const size\_t lda)  

$$\text{fscaln } A \leftarrow a \cdot A.$$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  

$$\text{fscal } B \leftarrow a \cdot A.$$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t ldx, `FFLAS_ELT` \*Y, const size\_t ldy)  

$$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$$
- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  

$$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  

$$\text{fadd} : \text{matrix addition.}$$
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  

$$\text{fsub} : \text{matrix subtraction.}$$
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  

$$\text{fsubin } C = C - B$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  

$$\text{fadd} : \text{matrix addition with scaling.}$$
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*faddin*

- template `INST_OR_DECL FFLAS_ELT * fgemv` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` TransA, const `size_t` M, const `size_t` N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*X, const `size_t` incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const `size_t` incY)

*finite prime FFLAS\_FIELD<FFLAS\_ELT> GEneral Matrix Vector multiplication.*

- template `INST_OR_DECL void fger` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const `size_t` incx, const `FFLAS_ELT` \*y, const `size_t` incy, `FFLAS_ELT` \*A, const `size_t` lda)

*fger: rank one update of a general matrix*

- template `INST_OR_DECL void ftrsv` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, int incX)

*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

- template `INST_OR_DECL void ftrsm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*B, const `size_t` ldb)

*ftrsm: TRIangular System solve with Matrix.*

- template `INST_OR_DECL void ftrmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*B, const `size_t` ldb)

*ftrmm: TRIangular Matrix Multiply.*

- template `INST_OR_DECL FFLAS_ELT * fgemmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fgemmm: Field GEneral Matrix Multiply.*

- template `INST_OR_DECL FFLAS_ELT * fgemmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq)
- template `INST_OR_DECL FFLAS_ELT * fgemmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive >` par)
- template `INST_OR_DECL FFLAS_ELT * fgemmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads >` par)
- template `INST_OR_DECL FFLAS_ELT * fsquare` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fsquare: Squares a matrix.*

- template<class Cut = `CuttingStrategy::Block`, class Strat = `StrategyParameter::Threads`>  
void `BlockCuts` (`size_t` &RBLOCKSIZE, `size_t` &CBLOCKSIZE, const `size_t` m, const `size_t` n, const `size_t` numthreads)
- template<> void `BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >` (`size_t` &RBLOCKSIZE, `size_t` &CBLOCKSIZE, const `size_t` m, const `size_t` n, const `size_t` numthreads)
- template<> void `BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >` (`size_t` &RBLOCKSIZE, `size_t` &CBLOCKSIZE, const `size_t` m, const `size_t` n, const `size_t` numthreads)
- template<> void `BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >` (`size_t` &RBLOCKSIZE, `size_t` &CBLOCKSIZE, const `size_t` m, const `size_t` n, const `size_t` grainsize)

- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`
- `template<class Field >  
void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class RandIter >  
void pfrand (const Field &F, RandIter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class Cut, class Param >  
Field::Element & fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field, class AlgoT, class FieldTrait >  
Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >  
Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >  
Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >  
Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`

- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, Al-`  
`goT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace >`  
`> &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda,`  
`const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, type-`  
`name Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`  
`CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
- `template<class Field , class AlgoT , class FieldTrait , class Cut >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda,`  
`const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, type-`  
`name Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`  
`CuttingStrategy::Row, Cut > > &H)`
- `void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)`
- `char * getArgumentsValue (int argc, char **argv, int i)`  
*Get the value of an argument and avoid core dump when no value was given after an argument.*
- `std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)`  
*writes the values of all arguments, preceded by the programName*
- `template<class Field >`  
`std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr`  
`A, size_t lda, FFLAS_FORMAT format, bool column_major)`  
*WriteMatrix: write a matrix to an output stream.*
- `void preamble (std::ifstream &if, FFLAS_FORMAT &format)`
- `template<class Field >`  
`Field::Element_ptr ReadMatrix (std::ifstream &if, Field &F, size_t &m, size_t &n, typename Field::Element_ptr`  
`&A, FFLAS_FORMAT format=FFlasAuto)`  
*ReadMatrix: read a matrix from an input stream.*
- `template<class Field >`  
`Field::Element_ptr ReadMatrix (const std::string &matrix_file, Field &F, size_t &m, size_t &n, typename`  
`Field::Element_ptr &A, FFLAS_FORMAT format=FFlasAuto)`  
*ReadMatrix: read a matrix from a file.*
- `template<class Field >`  
`void WriteMatrix (std::string &matrix_file, const Field &F, int m, int n, typename Field::ConstElement_ptr A,`  
`size_t lda, FFLAS_FORMAT format=FFlasDense, bool column_major=false)`  
*WriteMatrix: write a matrix to a file.*
- `std::ostream & WritePermutation (std::ostream &c, const size_t *P, size_t N)`  
*WritePermutation: write a permutation matrix to an output stream.*
- `template<class Element >`  
`bool alignable ()`
- `template<> bool alignable< Givaro::Integer * > ()`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const size_t n, const Alignment`  
`align=Alignment::DEFAULT)`
- `template<class Element >`  
`Element * fflas_new (const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<class Ptr , class ... Args>`  
`void fflas_delete (Ptr p, Args ... args)`

- void `prefetch` (const int64\_t \*)
- void `getTLBSize` (int &tlb)
- void `queryCacheSizes` (int &l1, int &l2, int &l3)
- int `queryL1CacheSize` ()
- int `queryTopLevelCacheSize` ()
- uint64\_t `getSeed` ()

### 15.1.1 Typedef Documentation

#### 15.1.1.1 Checker\_fgemm

```
using Checker_fgemm = FFLAS::Checker_Empty<Field>
```

#### 15.1.1.2 Checker\_ftrsm

```
using Checker_ftrsm = FFLAS::Checker_Empty<Field>
```

#### 15.1.1.3 ForceCheck\_fgemm

```
using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
```

#### 15.1.1.4 ForceCheck\_ftrsm

```
using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
```

#### 15.1.1.5 ZOSparseMatrix

```
using ZOSparseMatrix = std::true_type
```

#### 15.1.1.6 NotZOSparseMatrix

```
using NotZOSparseMatrix = std::false_type
```

#### 15.1.1.7 SimdSparseMatrix

```
using SimdSparseMatrix = std::true_type
```

#### 15.1.1.8 NoSimdSparseMatrix

```
using NoSimdSparseMatrix = std::false_type
```

#### 15.1.1.9 MKLSparseMatrixFormat

```
using MKLSparseMatrixFormat = std::true_type
```

#### 15.1.1.10 NotMKLSparseMatrixFormat

```
using NotMKLSparseMatrixFormat = std::false_type
```

#### 15.1.1.11 has\_plus

```
using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_impl<T> >::type
```

#### 15.1.1.12 has\_minus

```
using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_impl<T> >::type
```

#### 15.1.1.13 has\_equal

```
using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
std::is_copy_assignable<T> >::type
```

#### 15.1.1.14 has\_plus\_eq

```
using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_eq_impl<T> >::type
```

#### 15.1.1.15 has\_minus\_eq

```
using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_eq_impl<T> >::type
```

#### 15.1.1.16 has\_mul

```
using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>  
>::type
```

#### 15.1.1.17 has\_mul\_eq

```
using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_mul_eq_impl<T> >::type
```

#### 15.1.1.18 Timer

```
typedef Givaro::Timer Timer
```

#### 15.1.1.19 BaseTimer

```
typedef Givaro::BaseTimer BaseTimer
```

#### 15.1.1.20 UserTimer

```
typedef Givaro::UserTimer UserTimer
```

### 15.1.1.21 SysTimer

```
typedef Givaro::SysTimer SysTimer
```

## 15.1.2 Enumeration Type Documentation

### 15.1.2.1 FFLAS\_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

## Enumerator

FflasRowMajor	row major
FflasColMajor	col major

**15.1.2.2 FFLAS\_TRANSPOSE**enum [FFLAS\\_TRANSPOSE](#)

Is matrix transposed ?

## Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

**15.1.2.3 FFLAS\_UPLO**enum [FFLAS\\_UPLO](#)

Is triangular matrix's shape upper ?

## Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )
FflasLeftTri	Triangular matrix is Left triangular (if $j > n - i - 1$ then $T_{i,j} = 0$ )
FflasRightTri	Triangular matrix is Right triangular (if $j < n - i - 1$ then $T_{i,j} = 0$ )

**15.1.2.4 FFLAS\_DIAG**enum [FFLAS\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

## Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )

### 15.1.2.5 FFLAS\_SIDE

enum [FFLAS\\_SIDE](#)

On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.

### 15.1.2.6 FFLAS\_BASE

enum [FFLAS\\_BASE](#)

FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.

(deprecated, should not be used)

Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

### 15.1.2.7 number\_kind

enum [number\\_kind](#)

Enumerator

zero	
one	
mone	
other	

### 15.1.2.8 SparseMatrix\_t

enum [SparseMatrix\\_t](#) [strong]

## Enumerator

CSR	
CSR_ZO	
CSC	
CSC_ZO	
COO	
COO_ZO	
ELL	
ELL_ZO	
SELL	
SELL_ZO	
ELL_simd	
ELL_simd_ZO	
CSR_HYB	
HYB_ZO	

## 15.1.2.9 FFLAS\_FORMAT

enum [FFLAS\\_FORMAT](#)

## Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

## 15.1.3 Function Documentation

## 15.1.3.1 InfNorm()

```
Givaro::Integer FFLAS::InfNorm (
    const size_t M,
    const size_t N,
    const Givaro::Integer * A,
    const size_t lda ) [inline]
```

### 15.1.3.2 min3()

```
const T& FFLAS::min3 (
    const T & m,
    const T & n,
    const T & k )
```

### 15.1.3.3 max3()

```
const T& FFLAS::max3 (
    const T & m,
    const T & n,
    const T & k )
```

### 15.1.3.4 min4()

```
const T& FFLAS::min4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.5 max4()

```
const T& FFLAS::max4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.6 fadd() [1/8]

```
void FFLAS::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.7 faddin()** [1/5]

```
void FFLAS::faddin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.8 fsub()** [1/4]

```
void FFLAS::fsub (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.9 fsubin()** [1/3]

```
void FFLAS::fsubin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.10 fadd()** [2/8]

```
void FFLAS::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**Todo** optimise here

### 15.1.3.11 pfadd()

```
void FFLAS::pfadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

### 15.1.3.12 pfsub()

```
void FFLAS::pfsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

### 15.1.3.13 pfaddin()

```
void FFLAS::pfaddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

### 15.1.3.14 pfsubin()

```
void FFLAS::pfsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

**15.1.3.15 fadd()** [3/8]

```

void FFLAS::fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fadd : matrix addition.

Computes  $C = A + B$ .

**Parameters**

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

**15.1.3.16 fsub()** [2/4]

```

void FFLAS::fsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fsub : matrix subtraction.

Computes  $C = A - B$ .

**Parameters**

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols

## Parameters

<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

**15.1.3.17 faddin()** [2/5]

```
void FFLAS::faddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

faddin

**15.1.3.18 faddin()** [3/5]

```
void FFLAS::faddin (
    const Field & F,
    const FFLAS_UPLO uplo,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

fadding for symmetric matrices

**15.1.3.19 fsubin()** [2/3]

```
void FFLAS::fsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

fsubin  $C = C - B$

**15.1.3.20 fadd()** [4/8]

```

void FFLAS::fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \text{alpha } B$ .

**Parameters**

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>alpha</i>	some scalar
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

**15.1.3.21 fassign()** [1/10]

```

void FFLAS::fassign (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]

```

fassign :  $x \leftarrow y$ .

X is preallocated

**Todo** variant for triagular matrix

**Parameters**

	<i>F</i>	field
--	----------	-------

## Parameters

	$N$	size of the vectors
out	$X$	vector in $F$
	$incX$	stride of $X$
in	$Y$	vector in $F$
	$incY$	stride of $Y$

**15.1.3.22 fassign()** [2/10]

```
void FFLAS::fassign (
    const Givaro::Modular< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.23 fassign()** [3/10]

```
void FFLAS::fassign (
    const Givaro::ModularBalanced< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.24 fassign()** [4/10]

```
void FFLAS::fassign (
    const Givaro::ZRing< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.25 fassign()** [5/10]

```
void FFLAS::fassign (
    const Givaro::Modular< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.26 fassign()** [6/10]

```
void FFLAS::fassign (
    const Givaro::ModularBalanced< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.27 fassign()** [7/10]

```
void FFLAS::fassign (
    const Givaro::ZRing< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.28 fassign()** [8/10]

```
void FFLAS::fassign (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

fassign :  $A \leftarrow B$ .

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in F
$lda$	stride of A
$B$	vector in F

**15.1.3.29 faxpy()** [1/6]

```
void FFLAS::faxpy (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

	$F$	field
	$N$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in F
	$\text{incX}$	stride of X
in, out	$Y$	vector in F
	$\text{incY}$	stride of Y

**15.1.3.30 faxpy()** [2/6]

```
void FFLAS::faxpy (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.31 faxpy()** [3/6]

```
void FFLAS::faxpy (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

### 15.1.3.32 faxpy() [4/6]

```
void FFLAS::faxpy (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    typename Field::Element_ptr Y,
    const size_t ldy ) [inline]
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

#### Parameters

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$

### 15.1.3.33 fdot() [1/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

### 15.1.3.34 fdot() [2/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DelayedTag & MT ) [inline]
```

**15.1.3.35 fdot()** [3/11]

```
Givaro::DoubleDomain::Element FFLAS::fdot (
    const Givaro::DoubleDomain & ,
    const size_t N,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.36 fdot()** [4/11]

```
Givaro::FloatDomain::Element FFLAS::fdot (
    const Givaro::FloatDomain & ,
    const size_t N,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.37 fdot()** [5/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::ConvertTo< T > & MT ) [inline]
```

**15.1.3.38 fdot()** [6/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultBoundedTag & dbt ) [inline]
```

**15.1.3.39 fdot()** [7/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    const ParSeqHelper::Sequential seq ) [inline]
```

**15.1.3.40 fdot()** [8/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY ) [inline]
```

fdot: dot product  $x^T y$ .

**Parameters**

$F$	field
$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**15.1.3.41 fgemm()** [1/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
```

```

        const size_t ldc,
        MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > & H ) [inline]

```

#### 15.1.3.42 fgemm() [2/23]

```

FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Sequential > & H ) [inline]

```

#### 15.1.3.43 fgemm() [3/23]

```

FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads
>, ParSeqTrait > > & H ) [inline]

```

**15.1.3.44 fgemm()** [4/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Parallel< Cut, Param > > & H ) [inline]
```

**15.1.3.45 fgemm()** [5/23]

```
Givaro::Integer* FFLAS::fgemm (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
    ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.46 fgemm()** [6/23]

```
RNS::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
```

```

    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential
> & H ) [inline]

```

### 15.1.3.47 fgemm() [7/23]

```

RNS::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > & H ) [inline]

```

### 15.1.3.48 fgemm() [8/23]

```

Givaro::Integer* FFLAS::fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

**15.1.3.49 fgemm()** [9/23]

```
Givaro::Integer* FFLAS::fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.50 fgemm()** [10/23]

```
RecInt::ruint<K1>* FFLAS::fgemm (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * B,
    const size_t ldb,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.51 fgemm()** [11/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, ModeT > & H ) [inline]

```

### 15.1.3.52 fgemm() [12/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut,
Param > > & H ) [inline]

```

### 15.1.3.53 fgemm() [13/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFl
>, ParSeqHelper::Sequential > & H ) [inline]

```

**15.1.3.54 fgemm()** [14/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq ) [inline]
```

**15.1.3.55 fgemm()** [15/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.1.3.56 fgemm()** [16/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
```

```

typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc ) [inline]

```

**fgemm**: Field **G**eneral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

#### Parameters

<i>F</i>	field.
<i>ta</i>	if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	$C$ is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of $w$ .

#### Warning

$\alpha$  must be invertible

#### 15.1.3.57 fgemm() [17/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > & H ) [inline]

```

### 15.1.3.58 fgemm() [18/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H ) [inline]
```

### 15.1.3.59 fsquare() [1/6]

```
Field::Element_ptr FFLAS::fsquare (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a Field  $F$  Avoid the conversion of  $B$

#### Parameters

<i>ta</i>	if $ta == \text{FflasTrans}$ , $\text{op}(A) = A^T$ .
<i>F</i>	field
<i>n</i>	size of $A$
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size $n \times n$
<i>lda</i>	leading dimension of $A$
<i>C</i>	dense matrix of size $n \times n$
<i>ldc</i>	leading dimension of $C$

**Bug** why double ?

### 15.1.3.60 fsquare() [2/6]

```
double* FFLAS::fsquare (
    const Givaro::ModularBalanced< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

### 15.1.3.61 fsquare() [3/6]

```
float* FFLAS::fsquare (
    const Givaro::ModularBalanced< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

### 15.1.3.62 fsquare() [4/6]

```
double* FFLAS::fsquare (
    const Givaro::Modular< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

**15.1.3.63 fsquare()** [5/6]

```
float* FFLAS::fsquare (
    const Givaro::Modular< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

**15.1.3.64 fgemv()** [1/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]
```

**15.1.3.65 fgemv()** [2/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]
```

**15.1.3.66 fgemv()** [3/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.67 fgemv()** [4/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]
```

**15.1.3.68 fgemv()** [5/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .

## Parameters

	$F$	field
	$TransA$	if $TransA == FflaTrans$ then $op(A) = A^t$ .
	$M$	rows
	$N$	cols
	$alpha$	scalar
	$A$	dense matrix of size $M \times N$
	$lda$	leading dimension of $A$
	$X$	dense vector of size $N$
	$incX$	stride of $X$
	$beta$	scalar
out	$Y$	dense vector of size $M$
	$incY$	stride of $Y$

## 15.1.3.69 fgemv() [6/19]

```
Givaro::ZRing<int64_t>::Element_ptr FFLAS::fgemv (
    const Givaro::ZRing< int64_t > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * X,
    const size_t incX,
    const int64_t beta,
    int64_t * Y,
    const size_t incY,
    MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

## 15.1.3.70 fgemv() [7/19]

```
Givaro::DoubleDomain::Element_ptr FFLAS::fgemv (
    const Givaro::DoubleDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::DoubleDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::DoubleDomain::Element beta,
    Givaro::DoubleDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.71 fgemv()** [8/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]
```

**15.1.3.72 fgemv()** [9/19]

```
Givaro::FloatDomain::Element_ptr FFLAS::fgemv (
    const Givaro::FloatDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::FloatDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::FloatDomain::Element beta,
    Givaro::FloatDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.73 fgemv()** [10/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Parallel< Cut, Param > & parH )
```

**15.1.3.74 fgemv()** [11/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Sequential & seqH )
```

**15.1.3.75 fgemv()** [12/19]

```
FFPACK::rns_double::Element_ptr FFLAS::fgemv (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.76 fgemv()** [13/19]

```
FFPACK::rns_double::Element_ptr FFLAS::fgemv (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.77 fgemv()** [14/19]

```
Givaro::Integer* FFLAS::fgemv (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.78 fgemv()** [15/19]

```
Givaro::Integer* FFLAS::fgemv (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.79 fgemv()** [16/19]

```
RecInt::ruint<K1>* FFLAS::fgemv (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * X,
    const size_t incx,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * Y,
    const size_t incy,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.80 fger()** [1/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]

```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$

**Parameters**

	$F$	field
	$M$	rows
	$N$	cols
	$\alpha$	scalar
in, out	$A$	dense matrix of size MxN and leading dimension lda
	$lda$	leading dimension of A
	$x$	dense vector of size M
	$incx$	stride of X
	$y$	dense vector of size N
	$incy$	stride of Y

**15.1.3.81 fger()** [2/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

**15.1.3.82 fger()** [3/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, AnyTag > & H ) [inline]

```

**15.1.3.83 fger()** [4/12]

```

void FFLAS::fger (
    const Givaro::DoubleDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::DoubleDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.84 fger()** [5/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr x,
    const size_t incx,
    const typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

**15.1.3.85 fger()** [6/12]

```

void FFLAS::fger (
    const Givaro::FloatDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::FloatDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.86 fger()** [7/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]

```

**15.1.3.87 fger()** [8/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

**15.1.3.88 fger()** [9/12]

```

void FFLAS::fger (
    const Givaro::Modular< Givaro::Integer > & F,
    const size_t M,
    const size_t N,
    const typename Givaro::Integer alpha,
    typename Givaro::Integer * x,
    const size_t incx,
    typename Givaro::Integer * y,
    const size_t incy,
    typename Givaro::Integer * A,
    const size_t lda,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

**15.1.3.89 fger()** [10/12]

```

void FFLAS::fger (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSInteger< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.90 fger()** [11/12]

```

void FFLAS::fger (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSIntegerMod< RNS >::Element alpha,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > & H ) [inline]

```

**15.1.3.91 freduce()** [1/11]

```
void FFLAS::freduce (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

freduce  $x \leftarrow y \bmod F$ .

**Parameters**

$F$	field
$n$	size of the vectors
$Y$	vector of Element
$incY$	stride of Y
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.92 freduce()** [2/11]

```
void FFLAS::freduce (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

freduce  $x \leftarrow x \bmod F$ .

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.93 freduce\_constoverride()** [1/2]

```
void FFLAS::freduce_constoverride (
    const Field & F,
```

```

const size_t m,
typename Field::ConstElement_ptr A,
const size_t incX )

```

#### 15.1.3.94 finit() [1/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t n,
    ConstOtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

#### 15.1.3.95 finit() [2/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

finit Initializes X in F\$.

##### Parameters

$F$	field
$n$	size of the vectors
$X$	vector in F
$incX$	stride of X

#### 15.1.3.96 freduce() [3/11]

```

void FFLAS::freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

freduce  $A \leftarrow A \bmod F$ .

##### Parameters

$F$	field
-----	-------

## Parameters

$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.97 freduce()** [4/11]

```
void FFLAS::freduce (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

freduce for square symmetric matrices

**15.1.3.98 pfreduce()**

```
void FFLAS::pfreduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t numths )
```

**15.1.3.99 freduce()** [5/11]

```
void FFLAS::freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

freduce  $A \leftarrow B \bmod F$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$
$B$	matrix in $\text{Element}$
$ldb$	stride of $B$

**15.1.3.100 freduce\_constoverride()** [2/2]

```
void FFLAS::freduce_constoverride (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

**15.1.3.101 finit()** [3/8]

```
void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    const OtherElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{finit } A \leftarrow B \bmod F.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A
$B$	matrix in OtherElement
$ldb$	stride of B

**15.1.3.102 finit()** [4/8]

```
void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{finit}$  Initializes A in F\$.

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.103 freduce()** [6/11]

```
void FFLAS::freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    size_t inc ) [inline]
```

**15.1.3.104 freduce()** [7/11]

```
void FFLAS::freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    FFPACK::rns_double::Element_ptr A,
    size_t lda ) [inline]
```

**15.1.3.105 freivalds()**

```
bool FFLAS::freivalds (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C,
    const size_t ldc ) [inline]
```

freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.

Randomly Checks  $C = \alpha \text{op}(A) \times \text{op}(B)$

## Parameters

$F$	field.
$ta$	if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
$tb$	same for matrix B
$m$	see A
$n$	see B
$k$	see A
$\alpha$	scalar
$A$	$\text{op}(A)$ is $m \times k$
$B$	$\text{op}(B)$ is $k \times n$
$C$	$C$ is $m \times n$
$lda$	leading dimension of A
$ldb$	leading dimension of B
$ldc$	leading dimension of C

## 15.1.3.106 fscalin() [1/10]

```
void FFLAS::fscalin (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]
```

$\text{fscalin } x \leftarrow \alpha \cdot x.$

## Parameters

$F$	field
$n$	size of the vectors
$\alpha$	scalar
$X$	vector in $F$
$incX$	stride of X

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

## 15.1.3.107 fscale() [1/10]

```
void FFLAS::fscale (
    const Field & F,
```

```

const size_t n,
const typename Field::Element alpha,
typename Field::ConstElement_ptr X,
const size_t incX,
typename Field::Element_ptr Y,
const size_t incY ) [inline]

```

$\text{fscal } y \leftarrow \alpha \cdot x.$

#### Parameters

	$F$	field
	$n$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
out	$Y$	vector in $F$
	$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

#### 15.1.3.108 fscal() [2/10]

```

void FFLAS::fscal (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]

```

#### 15.1.3.109 fscal() [3/10]

```

void FFLAS::fscal (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]

```

**15.1.3.110 fscaln()** [2/10]

```
void FFLAS::fscaln (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.111 fscaln()** [3/10]

```
void FFLAS::fscaln (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.112 fscaln()** [4/10]

```
void FFLAS::fscaln (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

$\text{fscaln } A \leftarrow a \cdot A.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$\alpha$	homotecie scalar
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.113 fscl()** [4/10]

```
void FFLAS::fscl (
    const Field & F,
    const size_t m,
```

```

const size_t n,
const typename Field::Element alpha,
typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::Element_ptr B,
const size_t ldb ) [inline]

```

$\text{fscal } B \leftarrow a \cdot A.$

#### Parameters

	$F$	field
	$m$	number of rows
	$n$	number of cols
	$\alpha$	homotecie scalar
in	$A$	matrix in F
	$lda$	stride of A
out	$B$	matrix in F
	$ldb$	stride of B

#### 15.1.3.114 fscaln() [5/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t inc ) [inline]

```

#### 15.1.3.115 fscal() [5/10]

```

void FFLAS::fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]

```

#### 15.1.3.116 fscaln() [6/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]

```

**15.1.3.117 fscal()** [6/10]

```

void FFLAS::fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.118 fscaln()** [7/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha,
    typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    const size_t inc ) [inline]

```

**15.1.3.119 fscal()** [7/10]

```

void FFLAS::fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]

```

**15.1.3.120 fscaln()** [8/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]

```

**15.1.3.121 fscal()** [8/10]

```

void FFLAS::fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.122 fsyr2k()**

```

Field::Element_ptr FFLAS::fsyr2k (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

fsyr2k: Symmetric Rank 2K update

Computes the Lower or Upper triangular part of  $C = \alpha(A \times B^T + B \times A^T) + \beta C$  or  $C = \alpha(A^T \times B + B^T \times A) + \beta C$

**Parameters**

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha(A \times B^T + B \times A^T) + \beta C$ , else $C = \alpha(A^T \times B + B^T \times A) + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	A is $n \times k$ (FflasNoTrans) or A is $k \times n$ (FflasTrans)
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	C is $n \times n$
<i>ldc</i>	leading dimension of C

## Warning

$\alpha$  must be invertible

## 15.1.3.123 fsyrk() [1/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyrk: Symmetric Rank K update

Computes the Lower or Upper triangular part of  $C = \alpha A \times A^T + \beta C$  or  $C = \alpha A^T \times A + \beta C$

## Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	$A$ is $n \times k$ or $A$ is $k \times n$
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	$C$ is $n \times n$
<i>ldc</i>	leading dimension of C

## Warning

$\alpha$  must be invertible

## 15.1.3.124 fsyrk() [2/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
```

```

const FFLAS_TRANSPOSE trans,
const size_t N,
const size_t K,
const typename Field::Element alpha,
typename Field::ConstElement_ptr A,
const size_t lda,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const ParSeqHelper::Sequential seq ) [inline]

```

### 15.1.3.125 fsyrk() [3/16]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]

```

### 15.1.3.126 fsyrk() [4/16]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
>, ParSeqHelper::Sequential > & H ) [inline]

```

**15.1.3.127 fsyrk()** [5/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]
```

**15.1.3.128 fsyrk()** [6/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]
```

**15.1.3.129 fsyrk()** [7/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > & H ) [inline]
```

**15.1.3.130 fsyrk()** [8/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]
```

**15.1.3.131 fsyrk()** [9/16]

```
Givaro::FloatDomain::Element_ptr FFLAS::fsyrk (
    const Givaro::FloatDomain & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const Givaro::FloatDomain::Element alpha,
    Givaro::FloatDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::FloatDomain::Element beta,
    Givaro::FloatDomain::Element_ptr C,
    const size_t ldc,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.132 fsyrk()** [10/16]

```
Givaro::DoubleDomain::Element_ptr FFLAS::fsyrk (
    const Givaro::DoubleDomain & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const Givaro::DoubleDomain::Element alpha,
    Givaro::DoubleDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::DoubleDomain::Element beta,
    Givaro::DoubleDomain::Element_ptr C,
    const size_t ldc,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.133 fsyrk()** [11/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]
```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times D \times A^T + \beta C$  or  $C = \alpha A^T \times D \times A + \beta C$  where  $D$  is a diagonal matrix. Matrix  $A$  is updated into  $D \times A$  (if  $\text{trans} = \text{FflasTrans}$ ) or  $A \times D$  (if  $\text{trans} = \text{FflasNoTrans}$ ).

**Parameters**

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if $\text{ta} == \text{FflasNoTrans}$ then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	$A$ is $n \times k$ or $A$ is $k \times n$
<i>lda</i>	leading dimension of A
<i>D</i>	$D$ is $k \times k$ diagonal matrix, stored as a vector of k coefficients
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	$C$ is $n \times n$
<i>ldc</i>	leading dimension of C

**Warning**

$\alpha$  must be invertible

**15.1.3.134 fsyrk()** [12/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
```

```

const size_t N,
const size_t K,
const typename Field::Element alpha,
typename Field::Element_ptr A,
const size_t lda,
typename Field::ConstElement_ptr D,
const size_t incD,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const ParSeqHelper::Sequential seq,
const size_t threshold ) [inline]

```

### 15.1.3.135 fsyrk() [13/16]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold ) [inline]

```

### 15.1.3.136 fsyrk() [14/16]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const std::vector< bool > & twoBlock,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times \text{Delta} D \times A^T + \beta C$  or  $C = \alpha A^T \times \text{Delta} D \times A + \beta C$  where  $D$  is a diagonal matrix and  $\text{Delta}$  is a block diagonal with either 1 on the diagonal or 2x2 swap blocks Matrix  $A$  is updated into  $D \times A$  (if `trans = FflasTrans`) or  $A \times D$  (if `trans = FflasNoTrans`).

## Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \Delta D \times A^T + \beta C$ , else $C = \alpha A^T \Delta D \times A + \beta C$
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	$A$ is $n \times k$ or $A$ is $k \times n$
<i>lda</i>	leading dimension of A
<i>D</i>	$D$ is $k \times k$ diagonal matrix, stored as a vector of k coefficients
<i>twoBlocks</i>	a vector boolean indicating the beginning of each 2x2 blocs in Delta
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	$C$ is $n \times n$
<i>ldc</i>	leading dimension of C

## Warning

$\alpha$  must be invertible

## 15.1.3.137 computeS1S2()

```
void FFLAS::computeS1S2 (
    const Field & F,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element x,
    const typename Field::Element y,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr S,
    const size_t lds,
    typename Field::Element_ptr T,
    const size_t ldt,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

## 15.1.3.138 fsyrk() [15/16]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
```

```

    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H ) [inline]

```

### 15.1.3.139 fsyrk() [16/16]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, Mode > & H ) [inline]

```

### 15.1.3.140 fsyrk\_strassen() [1/2]

```

Field::Element_ptr FFLAS::fsyrk_strassen (
    const Field & F,
    const FFLAS_UPLO uplo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element y1,
    const typename Field::Element y2,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.1.3.141 ftrmm()** [1/3]

```

void FFLAS::ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]

```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

**Parameters**

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is implicitly unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

**15.1.3.142 ftrmm()** [2/3]

```

void FFLAS::ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,

```

```
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc ) [inline]
```

**ftmrm**: **T**riangular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .

#### Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is implicitly unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B
<i>beta</i>	scalar
<i>C</i>	matrix of size MxN
<i>ldc</i>	leading dim of C

#### 15.1.3.143 ftrsm() [1/9]

```
void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

#### 15.1.3.144 ftrsm() [2/9]

```
void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
```

```

const FFLAS_DIAG Diag,
const size_t M,
const size_t N,
const typename Field::Element alpha,
typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr B,
const size_t ldb,
const ParSeqHelper::Sequential & PSH ) [inline]

```

#### 15.1.3.145 ftrsm() [3/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Parallel< Cut, Param > & PSH ) [inline]

```

#### 15.1.3.146 ftrsm() [4/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Recursive, ParSeqTrait > & H ) [inline]

```

**15.1.3.147 ftrsm()** [5/9]

```

void FFLAS::ftrsm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * B,
    const size_t ldb ) [inline]

```

**15.1.3.148 cblas\_impstrsm()**

```

void FFLAS::cblas_impstrsm (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_SIDE Side,
    const enum FFLAS_UPLO Uplo,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_DIAG Diag,
    const int M,
    const int N,
    const FFPACK::rns_double_elt alpha,
    FFPACK::rns_double_elt_cstptr A,
    const int lda,
    FFPACK::rns_double_elt_ptr B,
    const int ldb ) [inline]

```

**15.1.3.149 ftrsv()** [1/2]

```

void FFLAS::ftrsv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX ) [inline]

```

ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

$F$	field
-----	-------

## Parameters

<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows or columns of A according to TransA
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

## 15.1.3.150 igemm\_()

```

void igemm_ (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    const size_t M,
    const size_t N,
    const size_t K,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * B,
    const size_t ldb,
    const int64_t beta,
    int64_t * C,
    const size_t ldc ) [inline]

```

## 15.1.3.151 finit() [5/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t n,
    const OtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

finit  $x \leftarrow y \bmod F$ .

## Parameters

<i>F</i>	field
<i>n</i>	size of the vectors
<i>Y</i>	vector of OtherElement
<i>incY</i>	stride of Y
<i>X</i>	vector in F
<i>incX</i>	stride of X

**Bug** use cblas\_(d)scal when possible

### 15.1.3.152 fconvert() [1/3]

```
void FFLAS::fconvert (
    const Field & F,
    const size_t n,
    OtherElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )
```

$\text{fconvert } x \leftarrow y \bmod F.$

#### Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of $F$
$incY$	stride of $Y$
$X$	vector in <code>OtherElement</code>
$incX$	stride of $X$

**Bug** use cblas\_(d)scal when possible

### 15.1.3.153 fnegin() [1/4]

```
void FFLAS::fnegin (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{fnegin } x \leftarrow -x.$

#### Parameters

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use cblas\_(d)scal when possible

**15.1.3.154 fneg()** [1/4]

```
void FFLAS::fneg (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{fneg } x \leftarrow -y.$

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**15.1.3.155 fzero()** [1/5]

```
void FFLAS::fzero (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{fzero} : A \leftarrow 0.$

**Parameters**

$F$	field
$n$	number of elements to zero
$X$	vector in $F$
$incX$	stride of $X$

**15.1.3.156 frand()** [1/2]

```
void FFLAS::frand (
    const Field & F,
    RandIter & G,
```

```

    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

`frand` :  $A \leftarrow \text{random}$ .

#### Parameters

$F$	field
$G$	randomiterator
$n$	number of elements to randomize
$X$	vector in $F$
$incX$	stride of $X$

#### 15.1.3.157 `fiszero()` [1/4]

```

bool FFLAS::fiszero (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX )

```

`fiszero` : test  $X = 0$ .

#### Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$

#### 15.1.3.158 `fequal()` [1/4]

```

bool FFLAS::fequal (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )

```

`fequal` : test  $X = Y$ .

#### Parameters

$F$	field
$n$	vector dimension

## Parameters

$X$	vector in $F$
$incX$	increment of $X$
$Y$	vector in $F$
$incY$	increment of $Y$

**15.1.3.159 faxpby()** [1/2]

```
void FFLAS::faxpby (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

## Parameters

	$F$	field
	$N$	size of the vectors
	$alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
	$beta$	scalar
in, out	$Y$	vector in $F$
	$incY$	stride of $Y$

## Note

this is a catlas function

**15.1.3.160 fdot()** [9/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    const ParSeqHelper::Parallel< Cut, Param > par )
```

**15.1.3.161 fswap()** [1/2]

```
void FFLAS::fswap (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY )
```

fswap:  $X \leftrightarrow Y$ .

**Bug** use cblas\_dswap when double

**Parameters**

$F$	field
$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**15.1.3.162 fzero()** [2/5]

```
void FFLAS::fzero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

fzero :  $A \leftarrow 0$ .

**Parameters**

$F$	field
$m$	number of rows to zero
$n$	number of cols to zero
$A$	matrix in $F$
$lda$	stride of $A$

**Warning**

may be buggy if Element is larger than int

**15.1.3.163 fzero()** [3/5]

```
void FFLAS::fzero (
    const Field & F,
    const FFLAS_UPLO shape,
    const FFLAS_DIAG diag,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

**fzero** :  $A \leftarrow 0$  for a triangular matrix.

**Parameters**

$F$	field
$shape$	shape of the triangular matrix
$m$	number of rows to zero
$n$	number of cols to zero
$A$	matrix in $F$
$lda$	stride of $A$

**Warning**

may be buggy if Element is larger than int

**15.1.3.164 frand()** [2/2]

```
void FFLAS::frand (
    const Field & F,
    RandIter & G,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

**frand** :  $A \leftarrow random$ .

**Parameters**

$F$	field
$G$	randomiterator
$m$	number of rows to randomize
$n$	number of cols to randomize
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.165 fequal()** [2/4]

```
bool FFLAS::fequal (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )
```

fequal : test  $A = B$ .

**Parameters**

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in F
$lda$	leading dimension of A
$B$	m x n matrix in F
$ldb$	leading dimension of B

**15.1.3.166 fiszero()** [2/4]

```
bool FFLAS::fiszero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

fiszero : test  $A = 0$ .

**Parameters**

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in F
$lda$	leading dimension of A

**15.1.3.167 fidentity()** [1/4]

```
void FFLAS::fidentity (
    const Field & F,
```

```

    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element & d )

```

creates a diagonal matrix

### 15.1.3.168 fidentity() [2/4]

```

void FFLAS::fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

creates a diagonal matrix

### 15.1.3.169 finit() [6/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

finit Initializes A in F\$.

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A

### 15.1.3.170 fconvert() [2/3]

```

void FFLAS::fconvert (
    const Field & F,
    const size_t m,
    const size_t n,
    OtherElement_ptr A,

```

```

const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb )

```

fconvert  $A \leftarrow B \bmod F$ .

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in OtherElement
$lda$	stride of A
$B$	matrix in F
$ldb$	stride of B

**Todo** check if  $n == lda$

#### 15.1.3.171 fnegin() [2/4]

```

void FFLAS::fnegin (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

fnegin  $A \leftarrow -A$ .

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A

**Todo** check if  $n == lda$

#### 15.1.3.172 fneg() [2/4]

```

void FFLAS::fneg (
    const Field & F,
    const size_t m,

```

```

const size_t n,
typename Field::ConstElement_ptr B,
const size_t ldb,
typename Field::Element_ptr A,
const size_t lda )

```

$\text{fneg } A \leftarrow -B.$

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**Todo** check if  $n == lda$

#### 15.1.3.173 faxpby() [2/2]

```

void FFLAS::faxpby (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t ldy )

```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

#### Parameters

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
	$\beta$	scalar
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$

#### Note

this is a catlas function

**15.1.3.174 fmove()** [1/2]

```
void FFLAS::fmove (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )
```

$\text{fmove} : A \leftarrow B \text{ and } B \leftarrow 0.$

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in $F$
$lda$	stride of $A$
$B$	matrix in $F$
$ldb$	stride of $B$

**15.1.3.175 bitsize()**

```
size_t FFLAS::bitsize (
    const Field & F,
    size_t M,
    size_t N,
    const typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

**bitsize:** Computes the largest bitsize of the matrix' coefficients.

If the matrix is over a modular prime field, it returns the bitsize of the largest element (in a bsolute value)

**Parameters**

$F$	field
$M$	rows
$N$	cols
$incX$	stride of $X$
$A$	a matrix of leading dimension $lda$ and size $M \times N$
$lda$	leading dimension of $A$

**15.1.3.176 bitsize< Givaro::ZRing< Givaro::Integer > >()**

```
size_t FFLAS::bitsize< Givaro::ZRing< Givaro::Integer > > (
```

```

const Givaro::ZRing< Givaro::Integer > & F,
size_t M,
size_t N,
const Givaro::Integer * A,
size_t lda ) [inline]

```

### 15.1.3.177 ftrmv()

```

void FFLAS::ftrmv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX )

```

ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$

#### Parameters

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows and columns of A
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^T$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit diagonal.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

### 15.1.3.178 ftrsm() [6/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )

```

ftsm: **T**Riangular **S**ystem solve with **M**atrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

#### Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

**Bug**  $\alpha$  must be non zero.

#### 15.1.3.179 fsyrk\_strassen() [2/2]

```
Field::Element_ptr FFLAS::fsyrk_strassen (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element y1,
    const typename Field::Element y2,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & H ) [inline]
```

#### 15.1.3.180 pfgemm() [1/7]

```
Field::Element_ptr FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
```

```

const size_t n,
const size_t k,
const typename Field::Element alpha,
typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
size_t numthreads = 0 )

```

### 15.1.3.181 pfgemm\_1D\_rec()

```

Field::Element* FFLAS::pfgemm_1D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )

```

### 15.1.3.182 pfgemm\_2D\_rec()

```

Field::Element* FFLAS::pfgemm_2D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )

```

**15.1.3.183 pfgemm\_3D\_rec()**

```
Field::Element* FFLAS::pfgemm_3D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )
```

**15.1.3.184 pfgemm\_3D\_rec2()**

```
Field::Element_ptr FFLAS::pfgemm_3D_rec2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )
```

**15.1.3.185 fgemm() [19/23]**

```
std::enable_if<!std::is_same<ModeTrait, ModeCategories::ConvertTo<ElementCategories::RNSElementTag>
>::value, typename Field::Element_ptr>::type FFLAS::fgemm (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
```

```

    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,
Param > > & H ) [inline]

```

### 15.1.3.186 ftrsm() [7/9]

```

Field::Element_ptr FFLAS::ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &
H ) [inline]

```

### 15.1.3.187 ftrsm() [8/9]

```

Field::Element_ptr FFLAS::ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > & H
) [inline]

```

**15.1.3.188 sparse\_delete()** [1/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO > & A ) [inline]
```

**15.1.3.189 sparse\_delete()** [2/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A ) [inline]
```

**15.1.3.190 sparse\_init()** [1/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.191 sparse\_init()** [2/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.192 sparse\_delete()** [3/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.193 sparse\_delete()** [4/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A ) [inline]
```

**15.1.3.194 sparse\_print()** [1/3]

```
std::ostream& FFLAS::sparse_print (
    std::ostream & os,
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.195 sparse\_init()** [3/16]

```
void FFLAS::sparse_init (
    const Givaro::Modular< Givaro::Integer > & F,
    Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.196 sparse\_init()** [4/16]

```
void FFLAS::sparse_init (
    const Givaro::ZRing< Givaro::Integer > & F,
    Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.197 sparse\_init()** [5/16]

```
void FFLAS::sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE >> & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR_ZO >
& A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.198 sparse\_init()** [6/16]

```

void FFLAS::sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE >> & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR > &
A,

    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.199 sparse\_init()** [7/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.200 sparse\_init()** [8/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.201 sparse\_delete()** [5/12]

```

void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A ) [inline]

```

**15.1.3.202 sparse\_init() [9/16]**

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.203 sparse\_delete() [6/12]**

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL > & A ) [inline]
```

**15.1.3.204 sparse\_delete() [7/12]**

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A ) [inline]
```

**15.1.3.205 sparse\_init() [10/16]**

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.206 sparse\_init() [11/16]**

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.207 sparse\_delete()** [8/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.208 sparse\_delete()** [9/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A ) [inline]
```

**15.1.3.209 sparse\_print()** [2/3]

```
void FFLAS::sparse_print (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.210 sparse\_init()** [12/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.211 sparse\_init()** [13/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.212 sparse\_delete()** [10/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A ) [inline]
```

**15.1.3.213 sparse\_init()** [14/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.214 operator<<()**

```
std::ostream& FFLAS::operator<< (
    std::ostream & os,
    const Sparse< _Field, SparseMatrix_t::HYB_ZO > & A )
```

**15.1.3.215 readSmsFormat()**

```
void FFLAS::readSmsFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.216 readSprFormat()**

```
void FFLAS::readSprFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.217** `getDataType()` [1/4]

```
std::enable_if<std::is_integral<T>::value,int> FFLAS::getDataType ( )
```

**15.1.3.218** `getDataType()` [2/4]

```
std::enable_if<std::is_floating_point<T>::value,int> FFLAS::getDataType ( )
```

**15.1.3.219** `getDataType()` [3/4]

```
std::enable_if<std::is_same<T,mpz_t>::value,int> FFLAS::getDataType ( )
```

**15.1.3.220** `getDataType()` [4/4]

```
int FFLAS::getDataType ( )
```

**15.1.3.221** `readMachineType()`

```
void FFLAS::readMachineType (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

**15.1.3.222** `readDnsFormat()`

```
void FFLAS::readDnsFormat (
    const std::string & path,
    const Field & F,
    index_t & rowdim,
    index_t & coldim,
    typename Field::Element_ptr & val )
```

**15.1.3.223 writeDnsFormat()**

```
void FFLAS::writeDnsFormat (
    const std::string & path,
    const Field & F,
    const index_t & rowdim,
    const index_t & coldim,
    typename Field::Element_ptr A,
    index_t ldA )
```

**15.1.3.224 fspmv()** [1/2]

```
void FFLAS::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ) [inline]
```

**15.1.3.225 sparse\_delete()** [11/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.226 sparse\_delete()** [12/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A ) [inline]
```

**15.1.3.227 sparse\_print()** [3/3]

```
void FFLAS::sparse_print (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.228 sparse\_init()** [15/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz,
    uint64_t sigma = 0 ) [inline]

```

**15.1.3.229 sparse\_init()** [16/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.230 computeDeviation()**

```

double FFLAS::computeDeviation (
    It begin,
    It end )

```

**15.1.3.231 getStat()**

```

StatsMatrix FFLAS::getStat (
    const Field & F,
    const index_t * row,
    const index_t * col,
    typename Field::ConstElement_ptr val,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz )

```

**15.1.3.232 fspmv()** [2/2]

```
void FFLAS::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    const typename Field::Element & beta,
    typename Field::Element_ptr y ) [inline]
```

**15.1.3.233 fspmm()**

```
void FFLAS::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    const typename Field::Element & beta,
    typename Field::Element_ptr y,
    int ldy ) [inline]
```

**15.1.3.234 maxCardinality()**

```
Field::Residu_t maxCardinality ( ) [inline]
```

**15.1.3.235 maxCardinality< Givaro::Modular< int64\_t > >()**

```
uint64_t FFLAS::maxCardinality< Givaro::Modular< int64_t > > ( ) [inline]
```

**15.1.3.236 maxCardinality< Givaro::Modular< int32\_t > >()**

```
uint32_t FFLAS::maxCardinality< Givaro::Modular< int32_t > > ( ) [inline]
```

**15.1.3.237 minCardinality()**

```
Field::Residu_t FFLAS::minCardinality ( ) [inline]
```

**15.1.3.238 fflas\_delete()** [1/3]

```
void FFLAS::fflas_delete (
    FFPACK::rns_double_elt_ptr A ) [inline]
```

**15.1.3.239 fflas\_delete()** [2/3]

```
void FFLAS::fflas_delete (
    FFPACK::rns_double_elt_cstptr A ) [inline]
```

**15.1.3.240 fflas\_new()** [1/7]

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

**15.1.3.241 fflas\_new()** [2/7]

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

**15.1.3.242 finit\_rns()** [1/2]

```
void FFLAS::finit_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

**15.1.3.243 finit\_trans\_rns()**

```
void FFLAS::finit_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

**15.1.3.244 fconvert\_rns() [1/2]**

```
void FFLAS::fconvert_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

**15.1.3.245 fconvert\_trans\_rns()**

```
void FFLAS::fconvert_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

**15.1.3.246 fflas\_new() [3/7]**

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

**15.1.3.247 fflas\_new()** [4/7]

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

**15.1.3.248 finit\_rns()** [2/2]

```
void FFLAS::finit_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A )
```

**15.1.3.249 fconvert\_rns()** [2/2]

```
void FFLAS::fconvert_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A )
```

**15.1.3.250 freduce()** [8/11]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

freduce  $x \leftarrow x \bmod F$ .

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use cblas\_(d)scal when possible

### 15.1.3.251 `freduce()` [9/11]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{freduce } x \leftarrow y \bmod F.$

#### Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of Element
$incY$	stride of Y
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

### 15.1.3.252 `finit()` [7/8]

```
template INST_OR_DECL void FFLAS::finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{finit } x \leftarrow y \bmod F.$

#### Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of OtherElement
$incY$	stride of Y
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

### 15.1.3.253 fconvert() [3/3]

```
template INST_OR_DECL void FFLAS::fconvert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

fconvert  $x \leftarrow y \bmod F$ .

#### Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of $F$
$incY$	stride of $Y$
$X$	vector in OtherElement
$incX$	stride of $X$

**Bug** use cblas\_(d)scal when possible

### 15.1.3.254 fnegin() [3/4]

```
template INST_OR_DECL void FFLAS::fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

fnegin  $x \leftarrow -x$ .

#### Parameters

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use cblas\_(d)scal when possible

**15.1.3.255 fneg()** [3/4]

```
template INST_OR_DECL void FFLAS::fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{fneg } x \leftarrow -y.$

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**15.1.3.256 fzero()** [4/5]

```
template INST_OR_DECL void FFLAS::fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{fzero} : A \leftarrow 0.$

**Parameters**

$F$	field
$n$	number of elements to zero
$X$	vector in $F$
$incX$	stride of $X$

**15.1.3.257 fiszero()** [3/4]

```
template INST_OR_DECL bool FFLAS::fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
```

```
const FFLAS_ELT * X,
const size_t incX )
```

fiszero : test  $X = 0$ .

#### Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$

#### 15.1.3.258 fequal() [3/4]

```
template INST_OR_DECL bool FFLAS::fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

fequal : test  $X = Y$ .

#### Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$
$Y$	vector in $F$
$incY$	increment of $Y$

#### 15.1.3.259 fassign() [9/10]

```
template INST_OR_DECL void FFLAS::fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

fassign :  $x \leftarrow y$ .

$X$  is preallocated

**Todo** variant for triangular matrix

## Parameters

	$F$	field
	$N$	size of the vectors
out	$X$	vector in $F$
	$incX$	stride of $X$
in	$Y$	vector in $F$
	$incY$	stride of $Y$

**15.1.3.260 fscaln()** [9/10]

```
template INST_OR_DECL void FFLAS::fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{fscaln } x \leftarrow \alpha \cdot x.$

## Parameters

$F$	field
$n$	size of the vectors
$alpha$	scalar
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

**15.1.3.261 fscal()** [9/10]

```
template INST_OR_DECL void FFLAS::fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{fscal } y \leftarrow \alpha \cdot x.$

## Parameters

	$F$	field
	$n$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
out	$Y$	vector in $F$
	$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

**15.1.3.262 faxpy()** [5/6]

```
template INST_OR_DECL void FFLAS::faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

## Parameters

	$F$	field
	$N$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
in, out	$Y$	vector in $F$
	$incY$	stride of $Y$

**15.1.3.263 fdot()** [10/11]

```
template INST_OR_DECL FFLAS_ELT FFLAS::fdot (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * X,
    const size_t incX,
```

```
const FFLAS_ELT * Y,
const size_t incY )
```

**faxpby**:  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .

#### Parameters

	$F$	field
	$N$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
	$\beta$	scalar
in, out	$Y$	vector in $F$
	$incY$	stride of $Y$

#### Note

this is a catlas function

**fdot**: dot product  $x^T y$ .

#### Parameters

$F$	field
$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

### 15.1.3.264 fswap() [2/2]

```
template INST_OR_DECL void FFLAS::fswap (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

**fswap**:  $X \leftrightarrow Y$ .

**Bug** use `cblas_dswap` when double

#### Parameters

$F$	field
-----	-------

## Parameters

$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**15.1.3.265 fadd()** [5/8]

```
template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.266 fsub()** [3/4]

```
template INST_OR_DECL void FFLAS::fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.267 faddin()** [4/5]

```
template INST_OR_DECL void FFLAS::faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.268 fadd()** [6/8]

```
template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.269 fassign()** [10/10]

```
template INST_OR_DECL void FFLAS::fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

fassign :  $A \leftarrow B$ .

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in F
$lda$	stride of A
$B$	vector in F
$ldb$	stride of B

**15.1.3.270 fzero()** [5/5]

```
template INST_OR_DECL void FFLAS::fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

fzero :  $A \leftarrow 0$ .

## Parameters

$F$	field
$m$	number of rows to zero
$n$	number of cols to zero
$A$	matrix in $F$
$lda$	stride of $A$

## Warning

may be buggy if Element is larger than int

**15.1.3.271 fequal() [4/4]**

```
template INST_OR_DECL bool FFLAS::fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb )
```

fequal : test  $A = B$ .

## Parameters

$F$	field
$m$	row dimension
$n$	column dimension
$A$	$m \times n$ matrix in $F$
$lda$	leading dimension of $A$
$B$	$m \times n$ matrix in $F$
$ldb$	leading dimension of $B$

**15.1.3.272 fiszero() [4/4]**

```
template INST_OR_DECL bool FFLAS::fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda )
```

fiszero : test  $A = 0$ .

## Parameters

$F$	field
$m$	row dimension
$n$	column dimension
$A$	$m \times n$ matrix in $F$
$lda$	leading dimension of $A$

**15.1.3.273 fidentity()** [3/4]

```
template INST_OR_DECL void FFLAS::fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT & d )
```

creates a diagonal matrix

**15.1.3.274 fidentity()** [4/4]

```
template INST_OR_DECL void FFLAS::fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

creates a diagonal matrix

**15.1.3.275 freduce()** [10/11]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

freduce  $A \leftarrow A \bmod F$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.276 freduce()** [11/11]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{freduce } A \leftarrow B \bmod F.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A
$B$	matrix in Element
$ldb$	stride of B

**15.1.3.277 finit()** [8/8]

```
template INST_OR_DECL void FFLAS::finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{finit } A \leftarrow B \bmod F.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A
$B$	matrix in F
$ldb$	stride of B

**15.1.3.278 fnegin()** [4/4]

```
template INST_OR_DECL void FFLAS::fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fnegin } A \leftarrow -A.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A

**15.1.3.279 fneg()** [4/4]

```
template INST_OR_DECL void FFLAS::fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fneg } A \leftarrow -B.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in F
$lda$	stride of A

**15.1.3.280 fscaln()** [10/10]

```
template INST_OR_DECL void FFLAS::fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
```

```
const FFLAS_ELT alpha,
FFLAS_ELT * A,
const size_t lda )
```

$\text{fscalin } A \leftarrow a \cdot A.$

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$\alpha$	homotecie scalar
$A$	matrix in F
$lda$	stride of A

#### 15.1.3.281 fscal() [10/10]

```
template INST_OR_DECL void FFLAS::fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{fscal } B \leftarrow a \cdot A.$

#### Parameters

	$F$	field
	$m$	number of rows
	$n$	number of cols
	$\alpha$	homotecie scalar
in	$A$	matrix in F
	$lda$	stride of A
out	$B$	matrix in F
	$ldb$	stride of B

#### 15.1.3.282 faxpy() [6/6]

```
template INST_OR_DECL void FFLAS::faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
```

```

const FFLAS_ELT * X,
const size_t ldx,
FFLAS_ELT * Y,
const size_t ldy )

```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

#### Parameters

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in F
	$ldx$	leading dimension of X
in, out	$Y$	vector in F
	$ldy$	leading dimension of Y

#### 15.1.3.283 fmove() [2/2]

```

template INST_OR_DECL void FFLAS::fmove (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )

```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

#### Parameters

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in F
	$ldx$	leading dimension of X
	$\beta$	scalar
in, out	$Y$	vector in F
	$ldy$	leading dimension of Y

#### Note

this is a catlas function

$\text{fmove} : A \leftarrow B \text{ and } B \leftarrow 0.$

## Parameters

<i>F</i>	field
<i>m</i>	number of rows to copy
<i>n</i>	number of cols to copy
<i>A</i>	matrix in F
<i>lda</i>	stride of A
<i>B</i>	vector in F
<i>ldb</i>	stride of B

**15.1.3.284 fadd()** [7/8]

```
template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fadd : matrix addition.

Computes  $C = A + B$ .

## Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

**15.1.3.285 fsub()** [4/4]

```
template INST_OR_DECL void FFLAS::fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
```

```

const size_t lda,
const FFLAS_ELT * B,
const size_t ldb,
FFLAS_ELT * C,
const size_t ldc )

```

fsub : matrix subtraction.

Computes  $C = A - B$ .

#### Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

#### 15.1.3.286 fsubin() [3/3]

```

template INST_OR_DECL void FFLAS::fsubin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

fsubin  $C = C - B$

#### 15.1.3.287 fadd() [8/8]

```

template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \text{alpha } B$ .

## Parameters

$F$	field
$M$	rows
$N$	cols
$A$	dense matrix of size $M \times N$
$lda$	leading dimension of A
$alpha$	some scalar
$B$	dense matrix of size $M \times N$
$ldb$	leading dimension of B
$C$	dense matrix of size $M \times N$
$ldc$	leading dimension of C

**15.1.3.288 faddin()** [5/5]

```
template INST_OR_DECL void FFLAS::faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

faddin

**15.1.3.289 fgemv()** [17/19]

```
template INST_OR_DECL FFLAS_ELT* FFLAS::fgemv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT beta,
    FFLAS_ELT * Y,
    const size_t incY )
```

finite prime FFLAS\_FIELD<FFLAS\_ELT> GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .

## Parameters

	$F$	field
	$TransA$	if $TransA == FflaTrans$ then $op(A) = A^t$ .
	$M$	rows
	$N$	cols
	$alpha$	scalar
	$A$	dense matrix of size $M \times N$
	$lda$	leading dimension of $A$
	$X$	dense vector of size $N$
	$incX$	stride of $X$
	$beta$	scalar
out	$Y$	dense vector of size $M$
	$incY$	stride of $Y$

## 15.1.3.290 fger() [12/12]

```
template INST_OR_DECL void FFLAS::fger (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * x,
    const size_t incx,
    const FFLAS_ELT * y,
    const size_t incy,
    FFLAS_ELT * A,
    const size_t lda )
```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$

## Parameters

	$F$	field
	$M$	rows
	$N$	cols
	$alpha$	scalar
in, out	$A$	dense matrix of size $M \times N$ and leading dimension $lda$
	$lda$	leading dimension of $A$
	$x$	dense vector of size $M$
	$incx$	stride of $X$
	$y$	dense vector of size $N$
	$incy$	stride of $Y$

**15.1.3.291 ftrsv()** [2/2]

```
template INST_OR_DECL void FFLAS::ftrsv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    int incX )
```

ftrsv: **TR**angular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows or columns of A according to TransA
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

**15.1.3.292 ftrsm()** [9/9]

```
template INST_OR_DECL void FFLAS::ftrsm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

ftrsm: **TR**angular System solve with **M**atrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

**Parameters**

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.

## Parameters

<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size $M \times N$
<i>ldb</i>	leading dim of B

**Bug**  $\alpha$  must be non zero.

15.1.3.293 `ftmm()` [3/3]

```
template INST_OR_DECL void FFLAS::ftmm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

`ftmm`: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

## Parameters

<i>F</i>	field
<i>Side</i>	if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is implicitly unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size $M \times N$
<i>ldb</i>	leading dim of B

## 15.1.3.294 fgemmm() [20/23]

```
template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )
```

fgemm: **F**ield **G**eneral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

## Parameters

<i>F</i>	field.
<i>ta</i>	if <code>ta==FflasTrans</code> then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	<i>C</i> is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of <i>w</i> .

## Warning

$\alpha$  must be invertible

## 15.1.3.295 fgemmm() [21/23]

```
template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb,
const size_t m,
const size_t n,
const size_t k,
const FFLAS_ELT alpha,
const FFLAS_ELT * A,
const size_t lda,
const FFLAS_ELT * B,
const size_t ldb,
const FFLAS_ELT beta,
FFLAS_ELT * C,
const size_t ldc,
const ParSeqHelper::Sequential seq )

```

### 15.1.3.296 fgemm() [22/23]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive
> par )

```

### 15.1.3.297 fgemm() [23/23]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,

```

```

        const size_t ldc,
        const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads
> par )

```

### 15.1.3.298 fsquare() [6/6]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fsquare (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )

```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a FFLAS\_FIELD <FFLAS\_ELT> F Avoid the conversion of B

#### Parameters

<i>ta</i>	if ta==FflasTrans, $\text{op}(A) = A^T$ .
<i>F</i>	field
<i>n</i>	size of A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size n×n
<i>lda</i>	leading dimension of A
<i>C</i>	dense matrix of size n×n
<i>ldc</i>	leading dimension of C

### 15.1.3.299 BlockCuts() [1/2]

```

void FFLAS::BlockCuts (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]

```

**15.1.3.300 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.301 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.302 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.303 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.304 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.305 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.306 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.307 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.308 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.309 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.310 BlockCuts()** [2/2]

```
void FFLAS::BlockCuts (
    size_t & rowBlockSize,
    size_t & colBlockSize,
    size_t & lastRBS,
    size_t & lastCBS,
    size_t & changeRBS,
    size_t & changeCBS,
    size_t & numRowsBlock,
    size_t & numColBlock,
    size_t m,
    size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.311 pfzero()**

```
void FFLAS::pfzero (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

**15.1.3.312 pfrand()**

```
void FFLAS::pfrand (
    const Field & F,
    RandIter & G,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

**15.1.3.313 fdot()** [11/11]

```
Field::Element& FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element & d,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.1.3.314 pfgemm() [2/7]**

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block,
StrategyParameter::Threads > > & H )
```

**15.1.3.315 pfgemm() [3/7]**

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDAdaptive > > & H )
```

**15.1.3.316 pfgemm() [4/7]**

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

const typename Field::ConstElement_ptr AA,
const size_t lda,
const typename Field::ConstElement_ptr BB,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element * C,
const size_t ldc,
MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoDAdaptive > > & H )

```

### 15.1.3.317 pfgemm() [5/7]

```

Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoD > > & H )

```

### 15.1.3.318 pfgemm() [6/7]

```

Field::Element_ptr FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeD > > & H )

```

**15.1.3.319 pfgemm()** [7/7]

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDInPlace > > & H )
```

**15.1.3.320 fgemv()** [18/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::Threads > > & H )
```

**15.1.3.321 fgemv()** [19/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
```

```

        const typename Field::Element beta,
        typename Field::Element_ptr Y,
        const size_t incY,
        MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row,
Cut > > & H )

```

#### 15.1.3.322 parseArguments()

```

void parseArguments (
    int argc,
    char ** argv,
    Argument * args,
    bool printDefaults = true )

```

#### 15.1.3.323 getArgumentValue()

```

char* FFLAS::getArgumentValue (
    int argc,
    char ** argv,
    int i )

```

Get the value of an argument and avoid core dump when no value was given after an argument.

##### Parameters

<i>argv</i>	argument value list
<i>i</i>	argument index

##### Returns

char\* argument value

#### 15.1.3.324 writeCommandString()

```

std::ostream& FFLAS::writeCommandString (
    std::ostream & os,
    Argument * args,
    const char * programName = nullptr )

```

writes the values of all arguments, preceded by the programName

**15.1.3.325 WriteMatrix()** [1/2]

```
std::ostream& FFLAS::WriteMatrix (
    std::ostream & c,
    const Field & F,
    size_t m,
    size_t n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format,
    bool column_major )
```

WriteMatrix: write a matrix to an output stream.

**Parameters**

<i>c</i>	output stream
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

**15.1.3.326 preamble()**

```
void FFLAS::preamble (
    std::ifstream & ifs,
    FFLAS_FORMAT & format ) [inline]
```

**15.1.3.327 ReadMatrix()** [1/2]

```
Field::Element_ptr FFLAS::ReadMatrix (
    std::ifstream & ifs,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto )
```

ReadMatrix: read a matrix from an input stream.

**Parameters**

	<i>ifs</i>	input stream
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
Generated by Doxygen <i>format</i>		input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

**15.1.3.328 ReadMatrix()** [2/2]

```
Field::Element_ptr FFLAS::ReadMatrix (
    const std::string & matrix_file,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto ) [inline]
```

ReadMatrix: read a matrix from a file.

**Parameters**

	<i>matrix_file</i>	filename
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
	<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

**15.1.3.329 WriteMatrix()** [2/2]

```
void FFLAS::WriteMatrix (
    std::string & matrix_file,
    const Field & F,
    int m,
    int n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format = FflasDense,
    bool column_major = false )
```

WriteMatrix: write a matrix to a file.

**Parameters**

<i>matrix_file</i>	file name
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

**15.1.3.330 WritePermutation()**

```
std::ostream& FFLAS::WritePermutation (
    std::ostream & c,
    const size_t * P,
    size_t N ) [inline]
```

WritePermutation: write a permutation matrix to an output stream.

**Parameters**

<i>c</i>	output stream
<i>P</i>	permutation
<i>N</i>	size of the permutation

**15.1.3.331 alignable()**

```
bool FFLAS::alignable ( ) [inline]
```

**15.1.3.332 alignable< Givaro::Integer \* >()**

```
bool FFLAS::alignable< Givaro::Integer * > ( ) [inline]
```

**15.1.3.333 fflas\_new() [5/7]**

```
Field::Element_ptr FFLAS::fflas_new (
    const Field & F,
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.334 fflas\_new() [6/7]**

```
Field::Element_ptr FFLAS::fflas_new (
    const Field & F,
    const size_t m,
    const size_t n,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.335 fflas\_new()** [7/7]

```
Element* FFLAS::fflas_new (
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.336 fflas\_delete()** [3/3]

```
void FFLAS::fflas_delete (
    Ptr p,
    Args ... args ) [inline]
```

**15.1.3.337 prefetch()**

```
void FFLAS::prefetch (
    const int64_t * ) [inline]
```

**15.1.3.338 getTLBSize()**

```
void FFLAS::getTLBSize (
    int & tlb ) [inline]
```

**15.1.3.339 queryCacheSizes()**

```
void FFLAS::queryCacheSizes (
    int & l1,
    int & l2,
    int & l3 ) [inline]
```

Queries and returns the cache sizes in Bytes of the L1, L2, and L3 data caches respectively

**15.1.3.340 queryL1CacheSize()**

```
int FFLAS::queryL1CacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L1 data cache

**15.1.3.341 queryTopLevelCacheSize()**

```
int FFLAS::queryTopLevelCacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L2 or L3 cache if this later is present

**15.1.3.342 getSeed()**

```
uint64_t FFLAS::getSeed ( )
```

**15.2 FFLAS::\_ftranspose\_impl Namespace Reference****Functions**

- template<size\_t bs, typename Field , typename BTSimd >  
void [not\\_inplace](#) (const [Field](#) &F, const BTSimd &BTS, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)
- template<size\_t bs, typename Field , typename BTSimd >  
void [square\\_inplace](#) (const [Field](#) &F, const BTSimd &BTS, const size\_t m, typename [Field::Element\\_ptr](#) A, const size\_t lda)
- template<size\_t bs, typename Field , typename BTSimd >  
void [nonsquare\\_inplace\\_v1](#) (const [Field](#) &F, const BTSimd &BTS, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A)
- template<size\_t bs, typename Field , typename BTSimd >  
void [nonsquare\\_inplace\\_v2](#) (const [Field](#) &F, const BTSimd &BTS, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A)

**15.2.1 Function Documentation****15.2.1.1 not\_inplace()**

```
void FFLAS::_ftranspose_impl::not_inplace (
    const Field & F,
    const BTSimd & BTS,
    const size_t m,
    const size_t n,
    typename Field::ConstElement\_ptr A,
    const size_t lda,
    typename Field::Element\_ptr B,
    const size_t ldb )
```

### 15.2.1.2 square\_inplace()

```
void FFLAS::_ftranspose_impl::square_inplace (
    const Field & F,
    const BTSimd & BTS,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t lda )
```

### 15.2.1.3 nonsquare\_inplace\_v1()

```
void FFLAS::_ftranspose_impl::nonsquare_inplace_v1 (
    const Field & F,
    const BTSimd & BTS,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A )
```

### 15.2.1.4 nonsquare\_inplace\_v2()

```
void FFLAS::_ftranspose_impl::nonsquare_inplace_v2 (
    const Field & F,
    const BTSimd & BTS,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A )
```

## 15.3 FFLAS::BLAS3 Namespace Reference

### Functions

- `template<class Field >`  
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE base, const size_t rec_level)`
- `template<class Field , class FieldTrait , class Strat , class Param >`  
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

- [illegible]

`Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const type-  
name Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field,  
MMHelperAlgo::Winograd, FieldTrait > &WH)`

### 15.3.1 Function Documentation

#### 15.3.1.1 Bini()

```
void FFLAS::BLAS3::Bini (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t kmax,
    const size_t w,
    const FFLAS_BASE base,
    const size_t rec_level ) [inline]
```

#### 15.3.1.2 WinoPar()

```
Field::Element_ptr FFLAS::BLAS3::WinoPar (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel<
Strat, Param > > & WH ) [inline]
```

### 15.3.1.3 Winograd()

```
void FFLAS::BLAS3::Winograd (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.3.1.4 WinogradAcc\_3\_23()

```
void FFLAS::BLAS3::WinogradAcc_3_23 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.3.1.5 WinogradAcc\_3\_21()

```
void FFLAS::BLAS3::WinogradAcc_3_21 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
```

```

typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.3.1.6 WinogradAcc\_2\_24()

```

void FFLAS::BLAS3::WinogradAcc_2_24 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.3.1.7 WinogradAcc\_2\_27()

```

void FFLAS::BLAS3::WinogradAcc_2_27 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.3.1.8 WinogradAcc\_LR()**

```

void FFLAS::BLAS3::WinogradAcc_LR (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.3.1.9 WinogradAcc\_R\_S()**

```

void FFLAS::BLAS3::WinogradAcc_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.3.1.10 WinogradAcc\_L\_S()**

```

void FFLAS::BLAS3::WinogradAcc_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,

```

```

const typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.3.1.11 Winograd\_LR\_S()

```

void FFLAS::BLAS3::Winograd_LR_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.3.1.12 Winograd\_L\_S()

```

void FFLAS::BLAS3::Winograd_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

### 15.3.1.13 Winograd\_R\_S()

```
void FFLAS::BLAS3::Winograd_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

## 15.4 FFLAS::csr\_hyb\_details Namespace Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)

## 15.5 FFLAS::CuttingStrategy Namespace Reference

### Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)

### Typedefs

- typedef [Row](#) [RNSModulus](#)

### 15.5.1 Typedef Documentation

#### 15.5.1.1 RNSModulus

```
typedef Row RNSModulus
```

## 15.6 FFLAS::details Namespace Reference

### Functions

- `template<class Field, bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`  
`FieldCategories::ModularTag)`
- `template<class Field, bool ADD>`  
`std::enable_if<!FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`  
`FieldCategories::ModularTag)`
- `template<class Field, bool ADD>`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, type-`  
`name Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`  
`FieldCategories::GenericTag)`
- `template<class Field, bool ADD>`  
`std::enable_if<!FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`  
`FieldCategories::UnparametricTag)`
- `template<class Field, bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`  
`FieldCategories::UnparametricTag)`
- `template<class Field>`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type faxpy (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC>`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<class Field>`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type freduce (const`  
`Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field>`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type freduce`  
`(const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename`  
`Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field, class FC>`  
`void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field, class FC>`  
`void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, type-`  
`name Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field>`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscaln (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field>`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscal (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`

- `template<class Field , class FC >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX, FC)`
- `template<class Field , class FC >`  
`void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<enum number_kind K>`  
`void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)`
- `template<size_t k, bool transpose>`  
`void pack_lhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)`
- `template<size_t k, bool transpose>`  
`void pack_rhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)`
- `void gebp (size_t rows, size_t cols, size_t depth, int64_t *C, size_t ldc, const int64_t *blockA, size_t lda, const int64_t *BlockB, size_t ldb, int64_t *BlockW)`
- `void BlockingFactor (size_t &m, size_t &n, size_t &k)`

## 15.6.1 Function Documentation

### 15.6.1.1 fadd() [1/5]

```
std::enable_if<FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↔
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )
```

**15.6.1.2 fadd() [2/5]**

```
std::enable_if<!FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )
```

**15.6.1.3 fadd() [3/5]**

```
void FFLAS::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::GenericTag )
```

**15.6.1.4 fadd() [4/5]**

```
std::enable_if<!FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

**15.6.1.5 fadd()** [5/5]

```
std::enable_if<FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

**15.6.1.6 faxpy()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::details::faxpy (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FieldCategories::ModularTag ) [inline]
```

**15.6.1.7 faxpy()** [2/2]

```
void FFLAS::details::faxpy (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FC ) [inline]
```

**15.6.1.8 freduce()** [1/4]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.6.1.9 freduce()** [2/4]

```
std::enable_if< FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.6.1.10 freduce()** [3/4]

```
void FFLAS::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

**15.6.1.11 freduce()** [4/4]

```
void FFLAS::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

**15.6.1.12 fscaln()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::details::fscaln (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.6.1.13 fscal()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS<
::details::fscal (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FieldCategories::ModularTag ) [inline]
```

**15.6.1.14 fscaln()** [2/2]

```
void FFLAS::details::fscaln (
    const Field & F,
    const size_t n,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FC ) [inline]
```

**15.6.1.15 fscal()** [2/2]

```
void FFLAS::details::fscal (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FC ) [inline]
```

**15.6.1.16 igebb44()**

```
void igebb44 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * b1A,
    const int64_t * b1B,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.6.1.17 igebb24()

```
void igebb24 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.6.1.18 igebb14()

```
void igebb14 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.6.1.19 igebb41()

```
void igebb41 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

bug ,B\_0 dans VEC\_MADD\_32 ?

bug ,B\_0 dans VEC\_MADD\_32 ?

### 15.6.1.20 igebb21()

```
void igebb21 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.6.1.21 igebb11()

```
void igebb11 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.6.1.22 igebp()

```
void igebp (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * blockA,
    size_t lda,
    const int64_t * blockB,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

### 15.6.1.23 pack\_lhs()

```
void pack_lhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

### 15.6.1.24 pack\_rhs()

```
void pack_rhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

### 15.6.1.25 gebp()

```
void FFLAS::details::gebp (
    size_t rows,
    size_t cols,
    size_t depth,
    int64_t * C,
    size_t ldc,
    const int64_t * blockA,
    size_t lda,
    const int64_t * BlockB,
    size_t ldb,
    int64_t * BlockW )
```

### 15.6.1.26 BlockingFactor()

```
void BlockingFactor (
    size_t & m,
    size_t & n,
    size_t & k ) [inline]
```

## 15.7 FFLAS::details\_spmv Namespace Reference

### Data Structures

- struct [Coo](#)

## 15.8 FFLAS::ElementCategories Namespace Reference

### Data Structures

- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: Givaro::reclnt.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: GMP.*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*

## 15.9 FFLAS::FieldCategories Namespace Reference

Traits and categories will need to be placed in a proper file later.

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*

### 15.9.1 Detailed Description

Traits and categories will need to be placed in a proper file later.

## 15.10 FFLAS::MMHelperAlgo Namespace Reference

### Data Structures

- struct [Auto](#)
- struct [Classic](#)
- struct [DivideAndConquer](#)
- struct [Winograd](#)
- struct [WinogradPar](#)
- struct [Bini](#)

## 15.11 FFLAS::ModeCategories Namespace Reference

Specifies the mode of action for an algorithm w.r.t.

### Data Structures

- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*

### 15.11.1 Detailed Description

Specifies the mode of action for an algorithm w.r.t.

its field

## 15.12 FFLAS::ParSeqHelper Namespace Reference

[ParSeqHelper](#) for both fgemm and ftrsm.

## Data Structures

- struct [Parallel](#)
- struct [Sequential](#)
- struct [Compose](#)

### 15.12.1 Detailed Description

[ParSeqHelper](#) for both fgemmm and ftrsm.

[ParSeqHelper](#) for both fgemmm and ftrsm

## 15.13 FFLAS::Protected Namespace Reference

### Data Structures

- class [AreEqual](#)
- class [AreEqual< X, X >](#)
- class [ftrsmLeftUpperNoTransNonUnit](#)  
*Computes the maximal size for delaying the modular reduction in a triangular system resolution.*
- class [ftrsmLeftUpperNoTransUnit](#)
- class [ftrsmLeftUpperTransNonUnit](#)
- class [ftrsmLeftUpperTransUnit](#)
- class [ftrsmLeftLowerNoTransNonUnit](#)
- class [ftrsmLeftLowerNoTransUnit](#)
- class [ftrsmLeftLowerTransNonUnit](#)
- class [ftrsmLeftLowerTransUnit](#)
- class [ftrsmRightUpperNoTransNonUnit](#)
- class [ftrsmRightUpperNoTransUnit](#)
- class [ftrsmRightUpperTransNonUnit](#)
- class [ftrsmRightUpperTransUnit](#)
- class [ftrsmRightLowerNoTransNonUnit](#)
- class [ftrsmRightLowerNoTransUnit](#)
- class [ftrsmRightLowerTransNonUnit](#)
- class [ftrsmRightLowerTransUnit](#)
- class [ftrmmLeftUpperNoTransNonUnit](#)
- class [ftrmmLeftUpperNoTransUnit](#)
- class [ftrmmLeftUpperTransNonUnit](#)
- class [ftrmmLeftUpperTransUnit](#)
- class [ftrmmLeftLowerNoTransNonUnit](#)
- class [ftrmmLeftLowerNoTransUnit](#)
- class [ftrmmLeftLowerTransNonUnit](#)
- class [ftrmmLeftLowerTransUnit](#)
- class [ftrmmRightUpperNoTransNonUnit](#)
- class [ftrmmRightUpperNoTransUnit](#)
- class [ftrmmRightUpperTransNonUnit](#)
- class [ftrmmRightUpperTransUnit](#)
- class [ftrmmRightLowerNoTransNonUnit](#)
- class [ftrmmRightLowerNoTransUnit](#)
- class [ftrmmRightLowerTransNonUnit](#)
- class [ftrmmRightLowerTransUnit](#)

## Functions

- `template<class Field >`  
`double computeFactorClassic (const Field &F)`
- `template<> double computeFactorClassic (const Givaro::ModularBalanced< double > &F)`
- `template<> double computeFactorClassic (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`  
`size_t DotProdBoundClassic (const Field &F, const typename Field::Element &beta)`
- `template<class Field >`  
`size_t TRSMBound (const Field &)`  
*TRSMBound.*
- `template<class Element >`  
`size_t TRSMBound (const Givaro::Modular< Element > &F)`  
*Specialization for positive modular representation over float.*
- `template<class Element >`  
`size_t TRSMBound (const Givaro::ModularBalanced< Element > &F)`  
*Specialization for balanced modular representation over double.*
- `template<class Field >`  
`int WinogradThreshold (const Field &F)`  
*Computes the number of recursive levels to perform.*
- `template<> int WinogradThreshold (const Givaro::Modular< float > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class Field , class FieldMode >`  
`void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class NewField , class Field , class FieldMode >`  
`Field::Element_ptr fgemm_convert (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`

- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait >`  
`&WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait >`  
`&WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Ele-`  
`ment &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT,`  
`ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element`  
`&Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, Par-`  
`SeqTrait > &WH)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename`  
`Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, Par-`  
`SeqTrait > &H)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag,`  
`ParSeqTrait > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename`  
`Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
- `template<typename FloatElement , class Field >`  
`Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const`  
`size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda,`  
`typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename`  
`Field::Element_ptr Y, const size_t incY)`
- `template<class FloatElement , class Field >`  
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t`  
`incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class NewField , class Field , class FieldMode >`  
`Field::Element_ptr fsyrk_convert (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE`  
`trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc,`  
`MMHelper< Field, MMHelperAlgo::Classic, FieldMode > &H)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const FFLAS_UPLO UpLo, const size_t N, const typename`  
`Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT,`  
`ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`  
`const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`

- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreAxyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreAxyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class DFE >`  
`size_t min\_types (const DFE &k)`
- `template<> size_t min\_types (const Reclnt::rint< 6 > &k)`
- `template<> size_t min\_types (const Reclnt::rint< 7 > &k)`
- `template<> size_t min\_types (const Reclnt::rint< 8 > &k)`
- `template<> size_t min\_types (const Reclnt::rint< 9 > &k)`
- `template<> size_t min\_types (const Reclnt::rint< 10 > &k)`
- `template<> size_t min\_types (const Givaro::Integer &k)`
- `template<class T >`  
`bool unfit (T x)`
- `template<> bool unfit (int64_t x)`
- `template<size_t K>`  
`bool unfit (Reclnt::rint< K > x)`
- `template<> bool unfit (Reclnt::rint< 6 > x)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
`void igemm\_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
`void igemm\_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `void igemm (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)`
- `template<class Field >`  
`void MatF2MatD\_Triangular (const Field &F, Givaro::DoubleDomain::Element\_ptr S, const size_t lds, typename Field::ConstElement\_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void MatF2MatFI\_Triangular (const Field &F, Givaro::FloatDomain::Element\_ptr S, const size_t lds, typename Field::ConstElement\_ptr const E, const size_t lde, const size_t m, const size_t n)`

## 15.13.1 Function Documentation

### 15.13.1.1 `computeFactorClassic()` [1/3]

```
double FFLAS::Protected::computeFactorClassic (
    const Field & F ) [inline]
```

### 15.13.1.2 `computeFactorClassic()` [2/3]

```
double FFLAS::Protected::computeFactorClassic (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.13.1.3 computeFactorClassic() [3/3]**

```
double FFLAS::Protected::computeFactorClassic (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.13.1.4 DotProdBoundClassic()**

```
size_t FFLAS::Protected::DotProdBoundClassic (
    const Field & F,
    const typename Field::Element & beta ) [inline]
```

**15.13.1.5 TRSMBound() [1/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Field & ) [inline]
```

TRSMBound.

computes the maximal size for delaying the modular reduction in a triangular system resolution

This is the default version over an arbitrary field. It is currently never used (the recursive algorithm is run until  $n=1$  in this case)

**Parameters**

$F$	Finite Field/Ring of the computation
-----	--------------------------------------

**15.13.1.6 TRSMBound() [2/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Givaro::Modular< Element > & F ) [inline]
```

Specialization for positive modular representation over float.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (p^{\{n_{\max}-1\}} + (p-2)^{\{n_{\max}-1\}}) < 2^{24}$  @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

**15.13.1.7 TRSMBound() [3/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Givaro::ModularBalanced< Element > & F ) [inline]
```

Specialization for balanced modular representation over double.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (((p+1)/2)^{\{n_{\max}-1\}}) < 2^{53}$

**Bibliography** • Dumas Giorgi Pernet 06, arXiv:cs/0601133

**15.13.1.8 WinogradThreshold()** [1/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Field & F ) [inline]
```

Computes the number of recursive levels to perform.

**Parameters**

<i>m</i>	the common dimension in the product AxB
----------	---

**15.13.1.9 WinogradThreshold()** [2/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::Modular< float > & F ) [inline]
```

**15.13.1.10 WinogradThreshold()** [3/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.13.1.11 WinogradThreshold()** [4/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.13.1.12 WinogradSteps()**

```
int FFLAS::Protected::WinogradSteps (
    const Field & F,
    const size_t & m ) [inline]
```

Computes the number of recursive levels to perform.

**Parameters**

<i>m</i>	the common dimension in the product AxB
----------	---

## 15.13.1.13 DynamicPeeling()

```

void FFLAS::Protected::DynamicPeeling (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmax ) [inline]

```

## 15.13.1.14 DynamicPeeling2()

```

void FFLAS::Protected::DynamicPeeling2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmax ) [inline]

```

### 15.13.1.15 WinogradCalc()

```
void FFLAS::Protected::WinogradCalc (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

### 15.13.1.16 fgemv\_convert()

```
Field::Element_ptr FFLAS::Protected::fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

### 15.13.1.17 NeedPreAddReduction() [1/2]

```
bool FFLAS::Protected::NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.13.1.18 NeedPreAddReduction()** [2/2]

```
bool FFLAS::Protected::NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.13.1.19 NeedPreSubReduction()** [1/2]

```
bool FFLAS::Protected::NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.13.1.20 NeedPreSubReduction()** [2/2]

```
bool FFLAS::Protected::NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.13.1.21 NeedDoublePreAddReduction()** [1/2]

```
bool FFLAS::Protected::NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.13.1.22 NeedDoublePreAddReduction() [2/2]**

```
bool FFLAS::Protected::NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.13.1.23 ScalAndReduce() [1/3]**

```
void FFLAS::Protected::ScalAndReduce (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

**15.13.1.24 ScalAndReduce() [2/3]**

```
void FFLAS::Protected::ScalAndReduce (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

**15.13.1.25 fsquareCommon()**

```
Field::Element_ptr FFLAS::Protected::fsquareCommon (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

**15.13.1.26 fgemv\_convert()**

```
Field::Element_ptr FFLAS::Protected::fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

**15.13.1.27 fger\_convert()**

```
void FFLAS::Protected::fger_convert (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

**15.13.1.28 fsyrk\_convert()**

```
Field::Element_ptr FFLAS::Protected::fsyrk_convert (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Classic, FieldMode > & H ) [inline]
```

**15.13.1.29 ScalAndReduce()** [3/3]

```

void FFLAS::Protected::ScalAndReduce (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]

```

**15.13.1.30 NeedPreScalReduction()** [1/2]

```

bool FFLAS::Protected::NeedPreScalReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    const Element & x,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]

```

**15.13.1.31 NeedPreScalReduction()** [2/2]

```

bool FFLAS::Protected::NeedPreScalReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    const Element & x,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]

```

**15.13.1.32 NeedPreAxyReduction()** [1/2]

```

bool FFLAS::Protected::NeedPreAxyReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    const Element & x,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]

```

**15.13.1.33 NeedPreAxyReduction() [2/2]**

```
bool FFLAS::Protected::NeedPreAxyReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    const Element & x,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.13.1.34 min\_types() [1/7]**

```
size_t FFLAS::Protected::min_types (
    const DFE & k ) [inline]
```

**15.13.1.35 min\_types() [2/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 6 > & k ) [inline]
```

**15.13.1.36 min\_types() [3/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 7 > & k ) [inline]
```

**15.13.1.37 min\_types() [4/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 8 > & k ) [inline]
```

**15.13.1.38 min\_types() [5/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 9 > & k ) [inline]
```

**15.13.1.39 min\_types() [6/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 10 > & k ) [inline]
```

**15.13.1.40 min\_types() [7/7]**

```
size_t FFLAS::Protected::min_types (
    const Givaro::Integer & k ) [inline]
```

**15.13.1.41 unfit() [1/4]**

```
bool FFLAS::Protected::unfit (
    T x ) [inline]
```

**15.13.1.42 unfit() [2/4]**

```
bool FFLAS::Protected::unfit (
    int64_t x ) [inline]
```

**15.13.1.43 unfit() [3/4]**

```
bool FFLAS::Protected::unfit (
    RecInt::rint< K > x ) [inline]
```

**15.13.1.44 unfit() [4/4]**

```
bool FFLAS::Protected::unfit (
    RecInt::rint< 6 > x ) [inline]
```

**15.13.1.45 igemm\_colmajor() [1/2]**

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.13.1.46 igemm\_colmajor() [2/2]**

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.13.1.47 igemm()**

```
void igemm (
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    const int64_t beta,
    int64_t * C,
    size_t ldc ) [inline]
```

**Todo** use primitive (no `Field()`) and specialise for int64.

**Todo** use primitive (no `Field()`) and specialise for int64.

### 15.13.1.48 MatF2MatD\_Triangular()

```
void FFLAS::Protected::MatF2MatD_Triangular (
    const Field & F,
    Givaro::DoubleDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

### 15.13.1.49 MatF2MatFl\_Triangular()

```
void FFLAS::Protected::MatF2MatFl_Triangular (
    const Field & F,
    Givaro::FloatDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

**Todo** do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)  
do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

## 15.14 FFLAS::sell\_details Namespace Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)

## 15.15 FFLAS::sparse\_details Namespace Reference

### Functions

- template<class Field >  
void [init\\_y](#) (const Field &F, const size\_t m, const typename Field::Element b, typename Field::Element\_ptr y)
- template<class Field >  
void [init\\_y](#) (const Field &F, const size\_t m, const size\_t n, const typename Field::Element b, typename Field::Element\_ptr y, const int ldy)
- template<class Field , class SM , class FC , class MZO >  
std::enable\_if< !(std::is\_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value)||std::is\_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value)>::type [fspmv\\_dispatch](#) (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FC fc, MZO mzo)

- `template<class Field , class SM , class FC , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmv\_dispatch (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FC fc, MZO mzo)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< lisSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< lisSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< lisSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value) || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`

- &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
  - `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value)>::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
  - `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`

## 15.15.1 Function Documentation

### 15.15.1.1 init\_y() [1/2]

```
void FFLAS::sparse_details::init_y (
    const Field & F,
    const size_t m,
    const typename Field::Element b,
    typename Field::Element_ptr y ) [inline]
```

**15.15.1.2 init\_y()** [2/2]

```
void FFLAS::sparse_details::init_y (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element b,
    typename Field::Element_ptr y,
    const int ldy ) [inline]
```

**15.15.1.3 fspmv\_dispatch()** [1/2]

```
std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fc,
    MZO mzo ) [inline]
```

**15.15.1.4 fspmv\_dispatch()** [2/2]

```
std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fc,
    MZO mzo ) [inline]
```

**15.15.1.5 fspmv()** [1/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.6 fspmv()** [2/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse\_details↵
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.7 fspmv()** [3/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse\_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.8 fspmv()** [4/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse\_details↵
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.9 fspmv()** [5/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse\_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.10 fspmv()** [6/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.11 fspmv()** [7/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details←
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.12 fspmv()** [8/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.13 fspmv()** [9/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```

**15.15.1.14 fspmm\_dispatch() [1/2]**

```
std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.15.1.15 fspmm\_dispatch() [2/2]**

```
std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.15.1.16 fspmm() [1/9]**

```
void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.17 fspmm() [2/9]**

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.18 fspmm() [3/9]**

```
std::enable_if<!support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.19 fspmm() [4/9]**

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.20 fspmm()** [5/9]

```
std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.21 fspmm()** [6/9]

```
void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.22 fspmm()** [7/9]

```
std::enable_if<support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.23 fspmm()** [8/9]

```

std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details↵
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]

```

**15.15.1.24 fspmm()** [9/9]

```

void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]

```

**15.15.1.25 pfspmm\_dispatch()** [1/2]

```

std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::Ma↵
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineIntT↵
::value)>::type FFLAS::sparse_details::pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.15.1.26 pfspmm\_dispatch()** [2/2]

```

std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value>::type FFLAS::sparse_details::pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.15.1.27 pfspmm()** [1/9]

```

void FFLAS::sparse_details::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]

```

**15.15.1.28 pfspmm()** [2/9]

```

std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.15.1.29 pfspmm()** [3/9]

```
std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details↵
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.30 pfspmm()** [4/9]

```
std::enable_if<support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details↵
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.31 pfspmm()** [5/9]

```
std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details↵
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.15.1.32 pfspmm()** [6/9]

```
void FFLAS::sparse_details::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.33 pfspmm()** [7/9]

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.34 pfspmm()** [8/9]

```
std::enable_if<!support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15.1.35 pfsppmm()** [9/9]

```

void FFLAS::sparse_details::pfsppmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]

```

**15.15.1.36 pfsppmv()** [1/6]

```

void FFLAS::sparse_details::pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::false_type ) [inline]

```

**15.15.1.37 pfsppmv()** [2/6]

```

void FFLAS::sparse_details::pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::false_type ) [inline]

```

**15.15.1.38 pfsppmv()** [3/6]

```

void FFLAS::sparse_details::pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::false_type ) [inline]

```

**15.15.1.39 pfspmv()** [4/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::true_type ) [inline]
```

**15.15.1.40 pfspmv()** [5/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::true_type ) [inline]
```

**15.15.1.41 pfspmv()** [6/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```

**15.15.1.42 fspmv()** [10/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>←
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

### 15.15.1.43 fspmv() [11/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

### 15.15.1.44 fspmv() [12/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

## 15.16 FFLAS::sparse\_details\_impl Namespace Reference

### Functions

- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::UnparametricTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_simd\_aligned (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_simd\_unaligned (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_one (const Field &F, const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)
- template<class Field >  
void fspmm\_mone (const Field &F, const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)

- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t`  
`blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t`  
`blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t`  
`blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t block_↵`  
`Size, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_↵`  
`_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_↵`  
`_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,`

- size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
- template<class Field >  
void [fspmm\\_one\\_simd\\_unaligned](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [fspmm\\_mone\\_simd\\_aligned](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [fspmm\\_mone\\_simd\\_unaligned](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
  - template<class Field >  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const int64\_t kmax)
  - template<class Field >  
void [fspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
  - template<class Field >  
void [fspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
  - template<class Field >  
void [fspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [fspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#))
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::GenericTag](#))
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#))
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, const int64\_t kmax)
  - template<class Field >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, const int64\_t kmax)
  - template<class Field >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_HYB](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field, class Func >`  
`void pfspm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field, class Func >`  
`void pfspm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`



- [illegible]

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 15.16.1 Function Documentation

**15.16.1.1 fspmm()** [1/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.2 fspmm()** [2/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.3 fspmm()** [3/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    const int64_t kmax ) [inline]
```

**15.16.1.4 fspmm\_simd\_aligned()** [1/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    const int64_t kmax ) [inline]
```

**15.16.1.5 fspmm\_simd\_unaligned()** [1/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.6 fspmm\_one()** [1/4]

```
void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.7 fspmm\_mone()** [1/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.8 fspmm\_one\_simd\_aligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.9 fspmm\_one\_simd\_unaligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.10 fspmm\_mone\_simd\_aligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.11 fspmm\_mone\_simd\_unaligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.12 fspmv()** [1/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.13 fspmv()** [2/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.14 fspmv()** [3/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.15 fspmv\_one()** [1/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.16 fspmv\_mone()** [1/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.17 fspmv\_one()** [2/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.18 fspmv\_mone()** [2/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.19 pfspmm()** [1/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.20 pfspmm()** [2/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.21 pfspmm()** [3/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.22 pfsppmm\_one() [1/2]**

```
void FFLAS::sparse_details_impl::pfsppmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.23 pfsppmm\_mone() [1/2]**

```
void FFLAS::sparse_details_impl::pfsppmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.24 pfsppmm\_one() [2/2]**

```
void FFLAS::sparse_details_impl::pfsppmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.25 pfsppmm\_mone() [2/2]**

```
void FFLAS::sparse_details_impl::pfsppmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.26 pfspmv()** [1/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.27 pfspmv\_task()**

```
void FFLAS::sparse_details_impl::pfspmv_task (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const index_t iStart,
    const index_t iStop,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.28 pfspmv()** [2/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.29 pfspmv()** [3/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.16.1.30 pfspmv\_one()** [1/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.31 pfspmv\_mone()** [1/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.32 pfspmv\_one()** [2/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.33 pfspmv\_mone()** [2/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.34 fspmm()** [4/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.35 fspmm()** [5/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    index_t blockSize,
    typename Field::ConstElement_ptr x_,
    index_t ldx,
    typename Field::Element_ptr y_,
    index_t ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.36 fspmm\_simd\_aligned()** [2/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.37 fspmm\_simd\_unaligned()** [2/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.38 fspmm()** [6/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.39 fspmm\_one() [2/4]**

```

void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.16.1.40 fspmm\_mone() [2/4]**

```

void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.16.1.41 fspmm\_one\_simd\_aligned() [2/3]**

```

void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.42 fspmm\_one\_simd\_unaligned() [2/3]**

```

void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.143 fspmm\_mone\_simd\_aligned()** [2/3]

```

void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.144 fspmm\_mone\_simd\_unaligned()** [2/3]

```

void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.145 fspmv()** [4/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.16.146 fspmv()** [5/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.47 fspmv()** [6/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.16.1.48 fspmv\_one()** [3/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.49 fspmv\_mone()** [3/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.50 fspmv\_one()** [4/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.51 fspmv\_mone()** [4/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.152 pfsppmm()** [4/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.16.153 pfsppmm()** [5/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.154 pfsppmm()** [6/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.155 pfsppmm()** [7/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.56 pfsppmm()** [8/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]

```

**15.16.1.57 pfsppmm()** [9/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.16.1.58 pfsppmv()** [4/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.16.1.59 pfsppmv()** [5/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.60 pfspmv()** [6/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.16.1.61 fspmm()** [7/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.62 fspmm()** [8/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.63 fspmm()** [9/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.64 fspmv()** [7/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.65 fspmv()** [8/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.66 fspmv()** [9/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.67 pfspmm()** [10/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.68 pfspmm()** [11/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.69 pfsppmm()** [12/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.70 pfsppmm()** [13/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.71 pfsppmm()** [14/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]
```

**15.16.1.72 pfsppmm()** [15/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.73 pfsppmm\_zo()** [1/2]

```
void FFLAS::sparse_details_impl::pfsppmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    Func && func ) [inline]
```

**15.16.1.74 pfsppmm\_zo()** [2/2]

```
void FFLAS::sparse_details_impl::pfsppmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    Func && func ) [inline]
```

**15.16.1.75 pfsppmv()** [7/18]

```
void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.76 pfsppmv()** [8/18]

```
void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.77 pfspmv()** [9/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.16.1.78 pfspmv\_one()** [3/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.79 pfspmv\_mone()** [3/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.80 pfspmv\_one()** [4/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.81 pfspmv\_mone()** [4/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.82 fspmm()** [10/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.83 fspmm()** [11/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.84 fspmm()** [12/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.16.1.85 fspmm\_mone()** [3/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.86 fspmm\_one() [3/4]**

```

void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.16.1.87 fspmm\_mone() [4/4]**

```

void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.88 fspmm\_one() [4/4]**

```

void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.89 fspmm\_one\_simd\_aligned() [3/3]**

```

void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.90 fspmm\_one\_simd\_unaligned()** [3/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.91 fspmm\_mone\_simd\_aligned()** [3/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.92 fspmm\_mone\_simd\_unaligned()** [3/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.93 fspmv()** [10/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.94 fspmv()** [11/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.95 fspmv()** [12/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.96 fspmv\_one()** [5/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.97 fspmv\_mone()** [5/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.98 fspmv\_one()** [6/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.99 fspmv\_mone()** [6/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.100 pfspmv()** [10/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.101 pfspmv()** [11/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.102 pfspmv()** [12/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.103 pfspmv\_one()** [5/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.104 pfspmv\_mone()** [5/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.105 pfspmv\_one()** [6/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.106 pfspmv\_mone()** [6/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.107 fspmv()** [13/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.108 fspmv\_simd()** [1/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.109 fspmv()** [14/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.110 fspmv\_simd()** [2/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.111 fspmv()** [15/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.112 fspmv\_one()** [7/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.113 fspmv\_mone()** [7/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.114 fspmv\_one()** [8/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.115 fspmv\_mone()** [8/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.116 fspmv\_one\_simd()** [1/2]

```
void FFLAS::sparse_details_impl::fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.117 fspmv\_mone\_simd()** [1/2]

```
void FFLAS::sparse_details_impl::fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.118 pfspmm()** [16/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.119 pfsppmm()** [17/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.120 pfsppmm()** [18/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]

```

**15.16.1.121 pfsppmv()** [13/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.16.1.122 pfsppmv()** [14/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16.1.123 pfspmv()** [15/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

**15.16.1.124 fspmm()** [13/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.125 fspmm()** [14/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.126 fspmm()** [15/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]
```

**15.16.1.127 fspmv()** [16/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.128 fspmv()** [17/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.129 fspmv()** [18/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

**15.16.1.130 pfspmv()** [16/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.131 pfspmv()** [17/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.132 pfspmv()** [18/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.16.1.133 pfspmv\_one()** [7/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.134 pfspmv\_mone()** [7/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.135 pfspmv\_one()** [8/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.136 pfspmv\_mone()** [8/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.137 fspmv()** [19/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.138 fspmv\_simd()** [3/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.139 fspmv()** [20/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.140 fspmv\_simd()** [4/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.141 fspmv()** [21/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.16.1.142 fspmv\_one()** [9/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.143 fspmv\_mone()** [9/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.16.1.144 fspmv\_one\_simd()** [2/2]

```
void FFLAS::sparse_details_impl::fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.145 fspmv\_mone\_simd()** [2/2]

```
void FFLAS::sparse_details_impl::fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.146 fspmv\_one()** [10/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.16.1.147 fspmv\_mone()** [10/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix\_t::SELL\_ZO > & A,
    typename Field::ConstElement\_ptr x_,
    typename Field::Element\_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.17 FFLAS::StrategyParameter Namespace Reference****Data Structures**

- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)

**15.18 FFLAS::StructureHelper Namespace Reference**

[StructureHelper](#) for ftrsm.

**Data Structures**

- struct [Recursive](#)
- struct [Iterative](#)
- struct [Hybrid](#)

**15.18.1 Detailed Description**

[StructureHelper](#) for ftrsm.

**15.19 FFLAS::vectorised Namespace Reference****Namespaces**

- [unswitch](#)

## Data Structures

- struct [HelperMod](#)
- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >

## Functions

- template<class SimdT, class Element, bool positive>  
std::enable\_if< [is\\_simd](#)< SimdT >::value, void >::type [VEC\\_ADD](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [addp](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n, Element p, T1 min\_, T2 max\_)
- template<class SimdT, class Element, bool positive>  
std::enable\_if< [is\\_simd](#)< SimdT >::value, void >::type [VEC\\_SUB](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [subp](#) (Element \*T, const Element \*TA, const Element \*TB, const size\_t n, const Element p, const T1 min\_, const T2 max\_)
- template<class Element >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [add](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n)
- template<class Element >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [sub](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [axpy](#) (const [Field](#) &F, const typename [Field::Element](#) a, typename [Field::ConstElement\\_ptr](#) X, typename [Field::Element\\_ptr](#) Y, const size\_t n)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [axpy](#) (const [Field](#) &F, const typename [Field::Element](#) a, typename [Field::ConstElement\\_ptr](#) X, typename [Field::Element\\_ptr](#) Y, const size\_t n, const size\_t incX, const size\_t incY)
- template<class T >  
std::enable\_if< ! std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<class T >  
std::enable\_if< std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<> [Givaro::Integer reduce](#) ([Givaro::Integer](#) A, [Givaro::Integer](#) B)
- float [reduce](#) (float A, float B, float invB, float min, float max)
- double [reduce](#) (double A, double B, double invB, double min, double max)
- int64\_t [reduce](#) (int64\_t A, int64\_t p, double invp, double min, double max, int64\_t pow50rem)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< [Field](#), [ElementCategories::MachineIntTag](#) > &H)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< [Field](#), [ElementCategories::MachineFloatTag](#) > &H)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< [Field](#), [ElementCategories::ArbitraryPrecIntTag](#) > &H)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [modp](#) (const [Field](#) &F, typename [Field::ConstElement\\_ptr](#) U, const size\_t &n, typename [Field::Element\\_ptr](#) T)

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type modp`  
`(const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename`  
`Field::Element_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY)`

## 15.19.1 Function Documentation

### 15.19.1.1 VEC\_ADD()

```
std::enable_if<is_simd<SimdT>::value, void>::type FFLAS::vectorised::VEC_ADD (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

### 15.19.1.2 addp()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::addp (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n,
    Element p,
    T1 min_,
    T2 max_ ) [inline]
```

### 15.19.1.3 VEC\_SUB()

```
std::enable_if<is_simd<SimdT>::value, void>::type FFLAS::vectorised::VEC_SUB (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

### 15.19.1.4 subp()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::subp (
    Element * T,
    const Element * TA,
    const Element * TB,
    const size_t n,
    const Element p,
    const T1 min_,
    const T2 max_ ) [inline]
```

### 15.19.1.5 add()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::add (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

### 15.19.1.6 sub()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::sub (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

**15.19.1.7 axpyp()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::axpyp (
    const Field & F,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    typename Field::Element_ptr Y,
    const size_t n ) [inline]
```

**15.19.1.8 axpyp()** [2/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::axpyp (
    const Field & F,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    typename Field::Element_ptr Y,
    const size_t n,
    const size_t incX,
    const size_t incY ) [inline]
```

**15.19.1.9 reduce()** [1/9]

```
std::enable_if< ! std::is_integral<T>::value, T>::type FFLAS::vectorised::reduce (
    T A,
    T B ) [inline]
```

**15.19.1.10 reduce()** [2/9]

```
std::enable_if< std::is_integral<T>::value, T>::type FFLAS::vectorised::reduce (
    T A,
    T B ) [inline]
```

**15.19.1.11 reduce()** [3/9]

```
Givaro::Integer FFLAS::vectorised::reduce (
    Givaro::Integer A,
    Givaro::Integer B ) [inline]
```

**15.19.1.12 reduce() [4/9]**

```
float FFLAS::vectorised::reduce (
    float A,
    float B,
    float invB,
    float min,
    float max ) [inline]
```

**15.19.1.13 reduce() [5/9]**

```
double FFLAS::vectorised::reduce (
    double A,
    double B,
    double invB,
    double min,
    double max ) [inline]
```

**15.19.1.14 reduce() [6/9]**

```
int64_t FFLAS::vectorised::reduce (
    int64_t A,
    int64_t p,
    double invp,
    double min,
    double max,
    int64_t pow50rem ) [inline]
```

**15.19.1.15 reduce() [7/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineIntTag > & H ) [inline]
```

**15.19.1.16 reduce() [8/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineFloatTag > & H ) [inline]
```

**15.19.1.17 reduce() [9/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H ) [inline]
```

**15.19.1.18 modp() [1/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T ) [inline]
```

**15.19.1.19 modp() [2/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T ) [inline]
```

**15.19.1.20 scalp() [1/3]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n ) [inline]
```

**15.19.1.21 scalp() [2/3]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX ) [inline]
```

15.19.1.22 `scalp()` [3/3]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    const size_t & incY ) [inline]
```

## 15.20 FFLAS::vectorised::unswitch Namespace Reference

## Functions

- template<class Field >  
std::enable\_if<!FFLAS::support\_simd\_mod< typename Field::Element >::value &&FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type axyp (const Field &F, const typename Field::Element a, typename Field::ConstElement\_ptr X, typename Field::Element\_ptr Y, const size\_t n, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type axyp (const Field &F, const typename Field::Element a, typename Field::ConstElement\_ptr X, typename Field::Element\_ptr Y, const size\_t n, const size\_t incX, const size\_t incY, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if<!FFLAS::support\_simd\_mod< typename Field::Element >::value &&FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size\_t &n, typename Field::Element\_ptr T, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size\_t &n, const size\_t &incX, typename Field::Element\_ptr T, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if<!FFLAS::support\_simd\_mod< typename Field::Element >::value &&FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size\_t n, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size\_t n, const size\_t &incX, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size\_t n, const size\_t &incX, const size\_t &incY, HelperMod< Field > &H)

## 15.20.1 Function Documentation

**15.20.1.1 axpyp()** [1/2]

```
std::enable_if<!FFLAS::support_simd_mod<typename Field::Element>::value && FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS::vectorised::unswitch::axpyp (
    const Field & F,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    typename Field::Element_ptr Y,
    const size_t n,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.2 axpyp()** [2/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::vectorised::unswitch::axpyp (
    const Field & F,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    typename Field::Element_ptr Y,
    const size_t n,
    const size_t incX,
    const size_t incY,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.3 modp()** [1/2]

```
std::enable_if<!FFLAS::support_simd_mod<typename Field::Element>::value && FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS::vectorised::unswitch::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.4 modp()** [2/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::vectorised::unswitch::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.5 scalp()** [1/3]

```
std::enable_if<!FFLAS::support_simd_mod<typename Field::Element>::value && FFLAS::support_fast_mod<typename
Field::Element>::value, void>::type FFLAS::vectorised::unswitch::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.6 scalp()** [2/3]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::vectorised::unswitch::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    HelperMod< Field > & H ) [inline]
```

**15.20.1.7 scalp()** [3/3]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::vectorised::unswitch::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    const size_t & incY,
    HelperMod< Field > & H ) [inline]
```

**15.21 FFPACK Namespace Reference**

Finite Field **PACK** Set of elimination based routines for dense linear algebra.

**Namespaces**

- [Protected](#)

## Data Structures

- class [CheckerImplem\\_charpoly](#)
- class [CheckerImplem\\_charpoly](#)< Givaro::ZRing< Givaro::Integer >, Polynomial >
- class [CheckerImplem\\_Det](#)
- class [CheckerImplem\\_invert](#)
- class [CheckerImplem\\_PLUQ](#)
- class [CharpolyFailed](#)
- class [callLUdivine\\_small](#)
- class [callLUdivine\\_small](#)< double >
- class [callLUdivine\\_small](#)< float >
- class [RNSInteger](#)
- class [RNSIntegerMod](#)
- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)
- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter](#)
- class [Failure](#)

*A precondition failed.*

## Typedefs

- template<class Field >  
using [Checker\\_PLUQ](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [Checker\\_Det](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [Checker\\_invert](#) = FFLAS::Checker\_Empty< Field >
- template<class Field , class Polynomial >  
using [Checker\\_charpoly](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [ForceCheck\\_PLUQ](#) = CheckerImplem\_PLUQ< Field >
- template<class Field >  
using [ForceCheck\\_Det](#) = CheckerImplem\_Det< Field >
- template<class Field >  
using [ForceCheck\\_invert](#) = CheckerImplem\_invert< Field >
- template<class Field , class Polynomial >  
using [ForceCheck\\_charpoly](#) = CheckerImplem\_charpoly< Field, Polynomial >

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class Field >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*

- `template<class Field >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field , class Cut , class Param >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field >`  
`void MonotonicApplyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t R)`  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- `template<class Field >`  
`void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t ldX, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldX, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*
- `template<class Field >`  
`void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)`  
*Compute the inverse of a triangular matrix.*
- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldL, typename Field::Element_ptr X, const size_t ldX)`
- `template<class Field >`  
`void ftrrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*Compute the product of two triangular matrices of opposite shape.*
- `template<class Field >`  
`void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)`  
*Solve a triangular system with a triangular right hand side of the same shape.*
- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PHelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)`  
*Compute the CUP or PLE factorization of the given matrix.*
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Row Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t [pRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field , class PSHelper >  
size\_t [RowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Compute the Reduced Column Echelon form of the input matrix in-place.*
- template<class Field >  
size\_t [pReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field , class PSHelper >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
size\_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Compute the Reduced Row Echelon form of the input matrix in-place.*
- template<class Field >  
size\_t [pReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field , class PSHelper >  
size\_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class Field >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class Field >  
[Field::Element\\_ptr Invert2](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class PolRing >  
std::list< typename PolRing::Element > & [CharPoly](#) (const PolRing &R, std::list< typename PolRing::Element > &charp, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=[FfpackAuto](#), const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))  
*Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & [CharPoly](#) (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=[FfpackAuto](#), const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))

*Compute the characteristic polynomial of the matrix A.*

- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N,`  
`typename PolRing::Domain_t::Element_ptr A, const size_t lda, const FFPACK_CHARPOLY_TAG Charp←`  
`Tag=FFpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`

*Compute the characteristic polynomial of the matrix A.*

- `template<class Field , class Polynomial >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda)`

*Compute the minimal polynomial of the matrix A.*

- `template<class Field , class Polynomial , class RandIter >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, RandIter &G)`

*Compute the minimal polynomial of the matrix A.*

- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::ConstElement_ptr v, const size_t incv)`

*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*

- `template<class Field >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)`

*Computes the rank of the given matrix using a PLUQ factorization.*

- `template<class Field >`  
`size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda,`  
`size_t numthreads=0)`
- `template<class Field , class PSHelper >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda,`  
`const PSHelper &psH)`
- `template<class Field >`  
`bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t`  
`lda)`

*Returns true if the given matrix is singular.*

- `template<class Field >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P=NULL, size_t *Q=NULL)`

*Returns the determinant of the given square matrix.*

- `template<class Field >`  
`Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field , class PSHelper >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda,`  
`typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)`

*Solves a linear system  $AX = b$  using PLUQ factorization.*

- `template<class Field , class PSHelper >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda,`  
`typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper`  
`&psH)`
- `template<class Field >`  
`Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda,`  
`typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t`  
`numthreads=0)`

- template<class Field >  
 \*void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- template<class Field >  
 size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- template<class Field >  
 size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the row rank profile of A.*
- template<class Field >  
 size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field, class PSHelper >  
 size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- template<class Field >  
 size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the column rank profile of A.*
- template<class Field >  
 size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field, class PSHelper >  
 size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- template<class Field >  
 size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*RowRankProfileSubmatrixIndices.*
- template<class Field >  
 size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- template<class Field >  
 size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class Field >  
 size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class Field >  
 void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*

- template<class Field >  
void [getTriangular](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr A, const size\_t lda)

*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*

- template<class Field >  
void [getEchelonForm](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- template<class Field >  
void [getEchelonForm](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename Field::Element\_ptr A, const size\_t lda, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*

- template<class Field >  
void [getEchelonTransform](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- template<class Field >  
void [getReducedEchelonForm](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*

- template<class Field >  
void [getReducedEchelonForm](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename Field::Element\_ptr A, const size\_t lda, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*

- template<class Field >  
void [getReducedEchelonTransform](#) (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt, const FFPACK\_LU\_TAG LuTag=FpackSlabRecursive)

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*

- template<class Field >  
size\_t [LTBruhatGen](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Q)

*LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.*

- template<class Field >  
void [getLTBruhatGen](#) (const Field &Fi, const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr R, const size\_t ldr)

*GetLTBruhatGen* This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.

- template<class Field >  
void [getLTBruhatGen](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt)

*GetLTBruhatGen* This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.

- size\_t [LTQSorder](#) (const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q)

*LTQSorder* This procedure computes the order of quasiseparability of a matrix.

- template<class Field >  
size\_t [CompressToBlockBiDiagonal](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_UPLO](#) Uplo, size\_t N, size\_t s, size\_t r, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) A, size\_t lda, typename [Field::Element\\_ptr](#) X, size\_t ldx, size\_t \*K, size\_t \*M, size\_t \*T)

*CompressToBlockBiDiagonal* This procedure compress a compact representation of a row echelon form or column echelon form.

- template<class Field >  
void [ExpandBlockBiDiagonalToBruhat](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_UPLO](#) Uplo, size\_t N, size\_t s, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda, typename [Field::Element\\_ptr](#) X, size\_t ldx, size\_t NbBlocks, size\_t \*K, size\_t \*M, size\_t \*T)

*ExpandBlockBiDiagonal* This procedure expand a compact representation of a row echelon form or column echelon form.

- void [Bruhat2EchelonPermutation](#) (size\_t N, size\_t R, const size\_t \*P, const size\_t \*Q, size\_t \*M)

*Bruhat2EchelonPermutation (N,R,P,Q)* Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.

- size\_t \* [TInverter](#) (size\_t \*T, size\_t r)

- template<class Field >  
void [ComputeRPermutation](#) (const [Field](#) &Fi, size\_t N, size\_t r, const size\_t \*P, const size\_t \*Q, size\_t \*R, size\_t \*MU, size\_t \*ML)

- template<class Field >  
void [productBruhatxTS](#) (const [Field](#) &Fi, size\_t N, size\_t s, size\_t r, const size\_t \*P, const size\_t \*Q, const typename [Field::Element\\_ptr](#) Xu, size\_t ldu, size\_t NbBlocksU, size\_t \*Ku, size\_t \*Tu, size\_t \*MU, const typename [Field::Element\\_ptr](#) XI, size\_t ldl, size\_t NbBlocksL, size\_t \*KL, size\_t \*TL, size\_t \*ML, typename [Field::Element\\_ptr](#) B, size\_t t, size\_t ldb, typename [Field::Element\\_ptr](#) C, size\_t ldc)

*productBruhatxTS* Comput the product between the CRE compact representation of a matrix A and B a tall matrix

- template<class Field >  
[Field::Element\\_ptr](#) [LQUPtoInverseOfFullRankMinor](#) (const [Field](#) &F, const size\_t rank, typename [Field::Element\\_ptr](#) A\_factors, const size\_t lda, const size\_t \*QtPointer, typename [Field::Element\\_ptr](#) X, const size\_t ldx)

*LQUPtoInverseOfFullRankMinor.*

- template<class Field >  
void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)

*Solve  $LX = B$  or  $XL = B$  in place.*

- template<class Field >  
void [solveLB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

- template<class Field >  
void [solveLB2](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

- size\_t \* [TInverter](#) (const size\_t \*T, size\_t r)

- template<class Field >  
void [ComputeRPermutation](#) (const [Field](#) &Fi, size\_t N, size\_t r, const size\_t \*P, const size\_t \*Q, size\_t \*R, const size\_t \*MU, const size\_t \*ML)

- `template<class Field >`  
`Field::Element_ptr expandLCRE (const Field &Fi, size_t N, size_t s, size_t r, size_t *R, size_t i, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tuinv, typename Field::ConstElement_ptr Xi, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tlinv, typename Field::Element_ptr CRE, size_t ldcre)`  
*Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.*
- `template<class Field >`  
`void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xi, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)`  
*Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B:  $C \leftarrow A \times B + \text{beta}C$ .*
- `template<class Field, class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)`
- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<class PSHelper >`  
`FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)`
- `template<class PSHelper >`  
`Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)`
- `template<class Field >`  
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin)`
- `template<class Field >`  
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Din, const size_t incDin, size_t *P, size_t BCThreshold)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename`

- [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, [FFLAS::ParSeqHelper::Sequential](#) seq, size\_t threshold)
- template<class Field, class Cut, class Param>  
bool [fsytrf\\_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par, size\_t threshold)
  - template<class Field>  
size\_t [fsytrf\\_RPM](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t threshold)
  - template<class Field>  
void [getTridiagonal](#) (const [Field](#) &F, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, size\_t \*P, typename [Field::Element\\_ptr](#) T, const size\_t ldt)
  - template<class Field>  
size\_t [LUdivine\\_gauss](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
  - template<class Field>  
size\_t [LUdivine\\_small](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
  - template<class Field>  
size\_t [LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
  - template<> size\_t [LUdivine](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
  - template<class Field>  
void [MonotonicCompress](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
  - template<class Field>  
void [MonotonicCompressMorePivots](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t rowstomove, const size\_t lenP)
  - template<class Field>  
void [MonotonicCompressCycles](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t lenP)
  - template<class Field>  
void [MonotonicExpand](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
  - template<class Field>  
void [applyP\\_block](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)
  - template<class Field>  
void [doApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
  - template<class Field>  
void [MatrixApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
  - template<class Field>  
void [MatrixApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, const [FFLAS::ParSeqHelper::Sequential](#) seq)

- `template<class Field , class Cut , class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field , class Cut , class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*
- `void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $MathP1 \times \text{Diag}(I_R, P2)$  where  $MathP1$  is a MathPermutation and  $P2$  a LAPACK permutation and store the result in  $MathP1$  as a MathPermutation format.*
- `void cyclic_shift_mathPerm (size_t *P, const size_t s)`
- `template<class Field >`  
`void cyclic_shift_row_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_row (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_row (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_col (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`size_t PLUQ_basecaseV3 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseCrout (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`

- `template<class Field >`  
`size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)`
- `template<class Cut , class Param >`  
`size_t PLUQ (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)`
- `template<class Field >`  
`void threads_fgemm (const size_t m, const size_t n, const size_t r, int nbthreads, size_t *W1, size_t *W2, size_t *W3, size_t gamma)`
- `template<class Field >`  
`void threads_ftsrm (const size_t m, const size_t n, int nbthreads, size_t *t1, size_t *t2)`
- `template<class Field >`  
`size_t PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &PSHelper)`
- `template<> rns_double_elt_ptr fflas_const_cast (rns_double_elt_cstptr x)`
- `template<> rns_double_elt_cstptr fflas_const_cast (rns_double_elt_ptr x)`
- `template<typename Base_t >`  
`void cyclic_shift_row_col (Base_t *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void cyclic_shift_row (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void cyclic_shift_col (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void applyP (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, FFLAS_ELT *A, const size_t lda, const size_t *P)`
- `template INST_OR_DECL void fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL FFLAS_ELT * fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL void ftrtri (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t threshold)`
- `template INST_OR_DECL void trinv_left (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *L, const size_t ldl, FFLAS_ELT *X, const size_t ldx)`
- `template INST_OR_DECL void ftrtrm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, FFLAS_ELT *A, const size_t lda)`
- `template INST_OR_DECL size_t PLUQ (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)`
- `template INST_OR_DECL size_t LUdivine (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag, const size_t cutoff)`
- `template INST_OR_DECL size_t LUdivine_small (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)`
- `template INST_OR_DECL size_t LUdivine_gauss (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)`

- template [INST\\_OR\\_DECL](#) size\_t [RowEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ReducedRowEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ReducedColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, [FFLAS\\_ELT](#) \*A, const size\_t lda, int &>nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, int &>nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert2](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, int &>nullity)
- template [INST\\_OR\\_DECL](#) std::list< Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > &[CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, std::list< Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const size\_t N, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const size\_t N, const [FFLAS\\_ELT](#) \*A, const size\_t lda)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MatVecMinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const size\_t N, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*V, const size\_t incv)
- template [INST\\_OR\\_DECL](#) size\_t [KrylovElim](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const size\_t deg, size\_t \*iterates, size\_t \*inviterates, const size\_t maxit, size\_t virt)
- template [INST\\_OR\\_DECL](#) size\_t [SpecRankProfile](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t deg, size\_t \*rankProfile)
- template [INST\\_OR\\_DECL](#) size\_t [Rank](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda)
- template [INST\\_OR\\_DECL](#) bool [IsSingular](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) & [Det](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) &det, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) & [Det](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) &det, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS::ParSeqHelper::Parallel](#)< [FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &parH, size\_t \*P, size\_t \*Q)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Solve](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*x, const int incx, const [FFLAS\\_ELT](#) \*b, const int incb)
- template [INST\\_OR\\_DECL](#) void [solveLB](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, [FFLAS\\_ELT](#) \*L, const size\_t ldl, const size\_t \*Q, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

- template `INST_OR_DECL` void `solveLB2` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, `FFLAS_ELT` \*L, const size\_t ldl, const size\_t \*Q, `FFLAS_ELT` \*B, const size\_t ldb)
- template `INST_OR_DECL` void `RandomNullSpaceVector` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, const size\_t incX)
- template `INST_OR_DECL` size\_t `NullSpaceBasis` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*&NS, size\_t &ldn, size\_t &NSdim)
- template `INST_OR_DECL` size\_t `RowRankProfile` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `ColumnRankProfile` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)
- template `INST_OR_DECL` size\_t `ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)
- template `INST_OR_DECL` size\_t `RowRankProfileSubmatrix` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, size\_t &R)
- template `INST_OR_DECL` size\_t `ColRankProfileSubmatrix` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, size\_t &R)
- template `INST_OR_DECL` void `getTriangular`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*T, const size\_t ldt, const bool OnlyNonZeroVectors)
- template `INST_OR_DECL` void `getTriangular`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, `FFLAS_ELT` \*A, const size\_t lda)
- template `INST_OR_DECL` void `getEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, `FFLAS_ELT` \*A, const size\_t lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*T, const size\_t ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, `FFLAS_ELT` \*A, const size\_t lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*T, const size\_t ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t rank, `FFLAS_ELT` \*A\_factors, const size\_t lda, const size\_t \*QtPointer, `FFLAS_ELT` \*X, const size\_t ldx)
- template<class T, class CT = const T>  
T `fflas_const_cast` (CT x)

- [Failure](#) & [failure](#) ()
- `template<class T >`  
`bool isOdd (const T &a)`
- `bool isOdd (const float &a)`
- `bool isOdd (const double &a)`
- `template<class Field , class RandIter >`  
`Field::Element\_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda, RandIter &G)`  
*Random non-zero Matrix.*
- `template<class Field , class RandIter >`  
`Field::Element\_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda)`  
*Random non-zero Matrix.*
- `template<class Field , class RandIter >`  
`Field::Element\_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda, RandIter &G)`  
*Random Matrix.*
- `template<class Field >`  
`Field::Element\_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda)`  
*Random Matrix.*
- `template<class Field , class RandIter >`  
`Field::Element\_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_DIAG Diag, bool nonsingular, typename Field::Element\_ptr A, size_t lda, RandIter &G)`  
*Random Triangular Matrix.*
- `template<class Field >`  
`Field::Element\_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_DIAG Diag, bool nonsingular, typename Field::Element\_ptr A, size_t lda)`  
*Random Triangular Matrix.*
- `size_t RandInt (size_t a, size_t b)`
- `template<class Field , class RandIter >`  
`Field::Element\_ptr RandomSymmetricMatrix (const Field &F, size_t n, bool nonsingular, typename Field::Element\_ptr A, size_t lda, RandIter &G)`  
*Random Symmetric Matrix.*
- `template<class Field , class RandIter >`  
`Field::Element\_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element\_ptr A, size_t lda, RandIter &G)`  
*Random Matrix with prescribed rank.*
- `template<class Field >`  
`Field::Element\_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element\_ptr A, size_t lda)`  
*Random Matrix with prescribed rank.*
- `size_t * RandomIndexSubset (size_t N, size_t R, size_t *P)`  
*Pick uniformly at random a sequence of R distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- `size_t * RandomPermutation (size_t N, size_t *P)`  
*Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.*
- `void RandomRankProfileMatrix (size_t M, size_t N, size_t R, size_t *rows, size_t *cols)`  
*Pick uniformly at random an R-subpermutation of dimension  $M \times N$  : a matrix with only R non-zeros equal to one, in a random rook placement.*
- `void swapval (size_t k, size_t N, size_t *P, size_t val)`
- `void RandomSymmetricRankProfileMatrix (size_t N, size_t R, size_t *rows, size_t *cols)`  
*Pick uniformly at random a symmetric R-subpermutation of dimension  $N \times N$  : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.*

- void [RandomLTQSRankProfileMatrix](#) (size\_t n, size\_t r, size\_t t, size\_t \*rows, size\_t \*cols)
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- template<class Field >  
[Field::Element\\_ptr RandomMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomSymmetricMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- template<class Field >  
[Field::Element\\_ptr RandomSymmetricMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomMatrixWithRankandRandomRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- template<class Field >  
[Field::Element\\_ptr RandomMatrixWithRankandRandomRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomSymmetricMatrixWithRankandRandomRPM](#) (const [Field](#) &F, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- template<class Field >  
[Field::Element\\_ptr RandomSymmetricMatrixWithRankandRandomRPM](#) (const [Field](#) &F, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- template<class Field >  
[Field::Element\\_ptr RandomMatrixWithDet](#) (const [Field](#) &F, size\_t n, const typename [Field::Element](#) d, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Matrix with prescribed det.*
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomMatrixWithDet](#) (const [Field](#) &F, size\_t n, const typename [Field::Element](#) d, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed det.*
- template<class Field , class Randlter >  
[Field::Element\\_ptr RandomLTQSMMatrixWithRankandQSorder](#) ([Field](#) &F, size\_t n, size\_t r, size\_t t, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)
- template<typename Field >  
[Field \\* chooseField](#) (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< int32\_t > \* [chooseField](#)< Givaro::ZRing< int32\_t > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< int64\_t > \* [chooseField](#)< Givaro::ZRing< int64\_t > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)

- `template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (Givaro::Integer q, uint64_t b, uint64_t seed)`

### 15.21.1 Detailed Description

**Finite Field PACK** Set of elimination based routines for dense linear algebra.

This namespace enlarges the set of BLAS routines of the class [FFLAS](#), with higher level routines based on elimination.

### 15.21.2 Typedef Documentation

#### 15.21.2.1 Checker\_PLUQ

```
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

#### 15.21.2.2 Checker\_Det

```
using Checker_Det = FFLAS::Checker_Empty<Field>
```

#### 15.21.2.3 Checker\_invert

```
using Checker_invert = FFLAS::Checker_Empty<Field>
```

#### 15.21.2.4 Checker\_charpoly

```
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

#### 15.21.2.5 ForceCheck\_PLUQ

```
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

### 15.21.2.6 ForceCheck\_Det

```
using ForceCheck_Det = CheckerImplem_Det<Field>
```

### 15.21.2.7 ForceCheck\_invert

```
using ForceCheck_invert = CheckerImplem_invert<Field>
```

### 15.21.2.8 ForceCheck\_charpoly

```
using ForceCheck_charpoly = CheckerImplem_charpoly<Field, Polynomial>
```

## 15.21.3 Function Documentation

### 15.21.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N ) [inline]
```

Conversion of a permutation from LAPACK format to Math format.

### 15.21.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N ) [inline]
```

Conversion of a permutation from Maths format to LAPACK format.

### 15.21.3.3 applyP() [1/4]

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P )
```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

## Parameters

<i>in, out</i>	<i>P1</i>	a LAPACK permutation of size N
	<i>P2</i>	a LAPACK permutation of size N-R

Applies a permutation P to the matrix A. Apply a permutation P, stored in the LAPACK format (a sequence of transpositions) between indices *ibeg* and *iend* of P to (*iend-ibeg*) vectors of size M stored in A (as column for NoTrans and rows for Trans). Side==[FFLAS::FflasLeft](#) for row permutation Side==[FFLAS::FflasRight](#) for a column permutation Trans==[FFLAS::FflasTrans](#) for the inverse permutation of P

## Parameters

<i>F</i>	base field
<i>Side</i>	decides if rows (FflasLeft) or columns (FflasRight) are permuted
<i>Trans</i>	decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)
<i>M</i>	size of the elements to permute
<i>ibeg</i>	first index to consider in P
<i>iend</i>	last index to consider in P
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	permutation in LAPACK format
<i>psh</i>	(optional): a sequential or parallel helper, to choose between sequential or parallel execution

## Warning

not sure the submatrix is still a permutation and the one we expect in all cases... examples for *iend*=2, *ibeg*=1 and *P*=[2,2,2]

15.21.3.4 **applyP()** [2/4]

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Sequential seq )
```

### 15.21.3.5 applyP() [3/4]

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par )
```

### 15.21.3.6 MonotonicApplyP()

```
void FFPACK::MonotonicApplyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t R )
```

Apply a R-monotonically increasing permutation P, to the matrix A.

MonotonicApplyP Apply a permutation defined by the first R entries of the vector P (the pivots).

The permutation represented by P is defined as follows:

- the first R values of P is a LAPACK representation (a sequence of transpositions)
- the remaining iend-ibeg-R values of the permutation are in a monotonically increasing progression Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

#### Parameters

<i>F</i>	base field
<i>Side</i>	selects if it is a row (FflasLeft) or column (FflasRight) permutation
<i>Trans</i>	inverse permutation (FflasTrans/NoTrans)
<i>M</i>	
<i>ibeg</i>	
<i>iend</i>	
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	LAPACK permuation
<i>R</i>	first values of P

The non pivot elements, are located in montonically increasing order.

### 15.21.3.7 fgetrs() [1/4]

```
void FFPACK::fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Solve the system  $AX = B$  or  $XA = B$ .

Solving using the PLUQ decomposition of A already computed inplace with PLUQ (FFLAS::FflasNonUnit). Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

#### Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of B
<i>N</i>	col dimension of B
<i>R</i>	rank of A
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	row permutation of the PLUQ decomposition of A
<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>B</i>	Right/Left hand side matrix. Initially stores B, finally stores the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

### 15.21.3.8 fgetrs() [2/4]

```
Field::Element_ptr FFPACK::fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
```

```

    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )

```

Solve the system  $A X = B$  or  $X A = B$ .

Solving using the PLUQ decomposition of A already computed inplace with PLUQ(FFLAS::FflasNonUnit). Version for A rectangular. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

#### Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
<i>R</i>	rank of A
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	row permutation of the PLUQ decomposition of A
<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>X</i>	solution matrix
<i>ldx</i>	leading dimension of X
<i>B</i>	Right/Left hand side matrix.
<i>ldb</i>	leading dimension of B
<i>info</i>	Succes of the computation: 0 if successfull, >0 if system is inconsistent

#### 15.21.3.9 fgesv() [1/4]

```

size_t FFPACK::fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )

```

Square system solver.

## Parameters

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of B
<i>N</i>	col dimension of B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

## Returns

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

## 15.21.3.10 fgesv() [2/4]

```
size_t FFPACK::fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )
```

Rectangular system solver.

## Parameters

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>X</i>	
<i>ldx</i>	
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

**Returns**

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for  $A$  square. If  $A$  is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

**15.21.3.11 ftrtri() [1/2]**

```
void FFPACK::ftrtri (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FTRTRI_THRESHOLD )
```

Compute the inverse of a triangular matrix.

**Parameters**

<i>F</i>	base field
<i>Uplo</i>	whether the matrix is upper or lower triangular
<i>Diag</i>	whether the matrix is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

**15.21.3.12 trinv\_left() [1/2]**

```
void FFPACK::trinv_left (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr L,
    const size_t ldl,
    typename Field::Element_ptr X,
    const size_t ldx )
```

**15.21.3.13 ftrtrm() [1/2]**

```
void FFPACK::ftrtrm (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
```

```

typename Field::Element_ptr A,
const size_t lda )

```

Compute the product of two triangular matrices of opposite shape.

Product UL or LU of the upper, resp lower triangular matrices U and L stored one above the other in the square matrix A.

#### Parameters

<i>F</i>	base field
<i>Side</i>	set to FflasLeft to compute the product UL, FflasRight to compute LU
<i>diag</i>	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

#### 15.21.3.14 ftrstr()

```

void FFPACK::ftrstr (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const FFLAS::FFLAS_DIAG diagB,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSTR_THRESHOLD )

```

Solve a triangular system with a triangular right hand side of the same shape.

#### Parameters

<i>F</i>	base field
<i>Side</i>	set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
<i>Uplo</i>	whether the matrix A is upper or lower triangular
<i>diag1</i>	whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)
<i>diag2</i>	whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	order of the input matrices
<i>A</i>	the input matrix to be inverted (U1 or L1)
<i>lda</i>	leading dimension of A
<i>B</i>	the input right hand side (U2 or L2)
<i>ldb</i>	leading dimension of B

## 15.21.3.15 ftrssyr2k()

```

void FFPACK::ftrssyr2k (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSSYR2K_THRESHOLD )

```

Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.

C is overwritten by B.

## Parameters

	<i>F</i>	base field
	<i>Side</i>	set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
	<i>Uplo</i>	whether the matrix A is upper or lower triangular
	<i>diagA</i>	whether the matrix A is unit diagonal (FflasUnit/NoUnit)
	<i>N</i>	order of the input matrices
	<i>A</i>	the input matrix
	<i>lda</i>	leading dimension of A
<i>in, out</i>	<i>B</i>	the input right hand side where the output is written
	<i>ldb</i>	leading dimension of B

## 15.21.3.16 fsytrf() [1/3]

```

bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )

```

Triangular factorization of symmetric matrices.

## Parameters

	<i>F</i>	The computation domain
	<i>UpLo</i>	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	<i>N</i>	order of the matrix A
<i>in, out</i>	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A

## Returns

false if the  $A$  does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix  $A$ :  $A = L \times D \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise.  $D$  is a diagonal matrix. The matrices  $L$  and  $U$  are unit diagonal lower (resp. upper) triangular and overwrite the input matrix  $A$ . The matrix  $D$  is stored on the diagonal of  $A$ , as the diagonal of  $L$  or  $U$  is known to be all ones. If  $A$  does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

## 15.21.3.17 fsytrf() [2/3]

```
bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Sequential seq,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

## 15.21.3.18 fsytrf() [3/3]

```
bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

## 15.21.3.19 fsytrf\_nonunit() [1/3]

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr D,
    const size_t incD,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

## Parameters

	$F$	The computation domain
	$UpLo$	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	$N$	order of the matrix $A$
in, out	$A$	input matrix
in, out	$D$	
	$lda$	leading dimension of $A$

## Returns

false if the  $A$  does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix  $A$ :  $A = L \times D_{inv} \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise.  $D$  is a diagonal matrix. The matrices  $L$  and  $U$  are lower (resp. upper) triangular and overwrite the input matrix  $A$ . The matrix  $D$  need to be stored separately, as the diagonal of  $L$  or  $U$  are not unit. If  $A$  does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

## 15.21.3.20 PLUQ() [1/6]

```
size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

Compute a PLUQ factorization of the given matrix.

Return its rank. The permutations P and Q are represented using LAPACK's convention.

## Parameters

$F$	base field
$Diag$	whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
$M$	matrix row dimension
$N$	matrix column dimension
$A$	input matrix
$lda$	leading dimension of $A$
$P$	the row permutation
$Q$	the column permutation

## Returns

the rank of  $A$

## Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013

## 15.21.3.21 pPLUQ()

```
size_t FFPACK::pPLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
```

```

    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q )

```

#### 15.21.3.22 PLUQ() [2/6]

```

size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Sequential & PSHelper,
    size_t BCThreshold = __FFLASFFPACK_PLUQ_THRESHOLD )

```

#### 15.21.3.23 PLUQ() [3/6]

```

size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper )

```

#### 15.21.3.24 LUdivine() [1/4]

```

size_t FFPACK::LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive,
    const size_t cutoff = __FFLASFFPACK_LUDIVINE_THRESHOLD )

```

Compute the CUP or PLE factorization of the given matrix.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

## Parameters

<i>F</i>	base field
<i>Diag</i>	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
<i>trans</i>	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
<i>M</i>	matrix row dimension
<i>N</i>	matrix column dimension
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	the factor of CUP or PLE
<i>Q</i>	a permutation indicating the pivot position in the echelon form C or E in its first r positions
<i>LuTag</i>	flag for setting the earling termination if the matrix is singular
<i>cutoff</i>	threshold to basecase

## Returns

the rank of A

## Bibliography

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

## 15.21.3.25 ColumnEchelonForm() [1/3]

```
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation  $A = [M \setminus V]$  such that  $AU = C$  is a column echelon decomposition of A, with  $U = P^T [V]$  and  $C = M + Q [I_r] [0 \text{ } I_{n-r}] [0]$  If LuTag == FfpackTileRecursive then  $A = [N \setminus V]$  such that the same holds with  $M = QN$

$Qt = Q^T$  If transform=false, the matrix V is not computed. See also test-colecheleon for an example of use

## Parameters

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
Generated by Doxygen	<i>P</i>	the column permutation
	<i>Qt</i>	the row position of the pivots in the echelon form
	<i>transform</i>	decides whether V is computed

### 15.21.3.26 pColumnEchelonForm()

```
size_t pColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

### 15.21.3.27 ColumnEchelonForm() [2/3]

```
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]
```

### 15.21.3.28 RowEchelonForm() [1/3]

```
size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Row Echelon form of the input matrix in-place.

If  $\text{LuTag} == \text{FfpackTileRecursive}$ , then after the computation  $A = [L \setminus M]$  such that  $XA = R$  is a row echelon decomposition of  $A$ , with  $X = [L \ 0]P$  and  $R = M + [I_r \ 0]Q^T$  [In-r] If  $\text{LuTag} == \text{FfpackTileRecursive}$  then  $A = [L \setminus N]$  such that the same holds with  $M = NQ^TQt = Q^T$  If  $\text{transform} = \text{false}$ , the matrix  $L$  is not computed. See also `test-rowechelon` for an example of use

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	the input matrix
	$lda$	leading dimension of A
	$P$	the row permutation
	$Qt$	the column position of the pivots in the echelon form
	<i>transform</i>	decides whether L is computed
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

## 15.21.3.29 pRowEchelonForm()

```

size_t pRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

## 15.21.3.30 RowEchelonForm() [2/3]

```

size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psh ) [inline]

```

**15.21.3.31 ReducedColumnEchelonForm()** [1/3]

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Column Echelon form of the input matrix in-place.

After the computation  $A = [V]$  such that  $AX = R$  is a reduced col echelon  $[M \ 0]$  decomposition of  $A$ , where  $X = P^T [V]$  and  $R = Q [I_r] [0 \ I_{n-r}] [M \ 0] Qt = Q^T$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

**Parameters**

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix
	$lda$	leading dimension of $A$
	$P$	the column permutation
	$Qt$	the row position of the pivots in the echelon form
	$transform$	decides whether $X$ is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

**15.21.3.32 pReducedColumnEchelonForm()**

```

size_t pReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

**15.21.3.33 ReducedColumnEchelonForm()** [2/3]

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & pSH ) [inline]

```

**15.21.3.34 ReducedRowEchelonForm()** [1/3]

```

size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Row Echelon form of the input matrix in-place.

After the computation  $A = [V1 \ M]$  such that  $X A = R$  is a reduced row echelon  $[V2 \ 0]$  decomposition of  $A$ , where  $X = [V1 \ 0] P$  and  $R = [I_r \ M] Q^T [V2 \ In-r] [0] Qt = Q^T$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

**Parameters**

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix
	$lda$	leading dimension of $A$
	$P$	the row permutation
	$Qt$	the column position of the pivots in the echelon form
	$transform$	decides whether $X$ is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

**15.21.3.35 pReducedRowEchelonForm()**

```

size_t pReducedRowEchelonForm (
    const Field & F,

```

```

const size_t M,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
size_t * P,
size_t * Qt,
const bool transform = false,
size_t numthreads = 0,
const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

### 15.21.3.36 ReducedRowEchelonForm() [2/3]

```

size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

### 15.21.3.37 Invert() [1/4]

```

Field::Element_ptr FFPACK::Invert (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    int & nullity )

```

Invert the given matrix in place or computes its nullity if it is singular.

An inplace  $2n^3$  algorithm is used.

#### Parameters

	$F$	The computation domain
	$M$	order of the matrix
in, out	$A$	input matrix ( $M \times M$ )
	$lda$	leading dimension of A
	$nullity$	dimension of the kernel of A

#### Returns

pointer to  $A$  and  $A \leftarrow A^{-1}$

**15.21.3.38 Invert()** [2/4]

```
Field::Element_ptr FFPACK::Invert (
    const Field & F,
    const size_t M,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

**Precondition**

X is preallocated and should be large enough to store the  $m \times m$  matrix A.

**Parameters**

	<i>F</i>	The computation domain
	<i>M</i>	order of the matrix
in	<i>A</i>	input matrix ( $M \times M$ )
	<i>lda</i>	leading dimension of A
out	<i>X</i>	this is the inverse of A if A is invertible (non NULL and nullity = 0). It is untouched otherwise.
	<i>ldx</i>	leading dimension of X
	<i>nullity</i>	dimension of the kernel of A

**Returns**

pointer to  $X = A^{-1}$

**15.21.3.39 Invert2()** [1/2]

```
Field::Element_ptr FFPACK::Invert2 (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

An  $2n^3f$  algorithm is used. This routine can be % faster than [FFPACK::Invert](#) but is not totally inplace.

**Precondition**

$X$  is preallocated and should be large enough to store the  $m \times m$  matrix  $A$ .

**Warning**

$A$  is overwritten here !

**Bug** not tested.

**Parameters**

	$F$	the computation domain
	$M$	order of the matrix
in, out	$A$	input matrix ( $M \times M$ ). On output, $A$ is modified and represents a "psycological" factorisation LU.
	$lda$	leading dimension of $A$
out	$X$	this is the inverse of $A$ if $A$ is invertible (non NULL and nullity = 0). It is untouched otherwise.
	$ldx$	leading dimension of $X$
	$nullity$	dimension of the kernel of $A$

**Returns**

pointer to  $X = A^{-1}$

**Todo** this init is not all necessary (done after ftrtri)

**15.21.3.40 CharPoly() [1/8]**

```
std::list< typename PolRing::Element > & CharPoly (
    const PolRing & R,
    std::list< typename PolRing::Element > & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]
```

Compute the characteristic polynomial of the matrix  $A$ .

**Parameters**

	$R$	the polynomial ring of charp (contains the base field)
out	$charp$	the characteristic polynomial of $A$ as a list of factors
	$N$	order of the matrix $A$
in	$A$	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	$lda$	leading dimension of $A$
	$CharpTag$	the algorithmic variant
	$G$	a random iterator (required for the randomized variants LUKrylov and ArithProg)

## 15.21.3.41 CharPoly() [2/8]

```

PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

## Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of as a single polynomial
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant
	<i>G</i>	a random iterator (required for the randomized variants LUKrylov and ArithProg)

## 15.21.3.42 CharPoly() [3/8]

```

PolRing::Element& FFPACK::CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

## Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of as a single polynomial
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant

**15.21.3.43 MinPoly()** [1/4]

```
Polynomial& FFPACK::MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector:  $(v, Av, \dots, A^kv)$

**Parameters**

	$F$	the base field
out	$minP$	the minimal polynomial of A
	$N$	order of the matrix A
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of A

**15.21.3.44 MinPoly()** [2/4]

```
Polynomial& FFPACK::MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    RandIter & G )
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector:  $(v, Av, \dots, A^kv)$

**Parameters**

	$F$	the base field
out	$minP$	the minimal polynomial of A
	$N$	order of the matrix A
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of A
	$G$	a random iterator

**15.21.3.45 MatVecMinPoly()** [1/2]

```
Polynomial& FFPACK::MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr v,
    const size_t incv )
```

Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .

**Parameters**

	$F$	the base field
out	$minP$	the minimal polynomial of A and v
	$N$	order of the matrix A
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of A
	$K$	an $N \times (N + 1)$ matrix containing the vector v on its first row
	$ldk$	leading dimension of K
	$P$	[out] (optional) the permutation used in the elimination of the Krylov matrix K

**15.21.3.46 Rank()** [1/3]

```
size_t FFPACK::Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Computes the rank of the given matrix using a PLUQ factorization.

The input matrix is modified.

**Parameters**

	$F$	base field
	$M$	row dimension of the matrix
	$N$	column dimension of the matrix
in	$A$	input matrix
	$lda$	leading dimension of A
	$psH$	(optional) a ParSeqHelper to choose between sequential and parallel execution

**15.21.3.47 pRank()**

```
size_t FFPACK::pRank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0 )
```

**15.21.3.48 Rank()** [2/3]

```
size_t FFPACK::Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & pSH )
```

**15.21.3.49 IsSingular()** [1/2]

```
bool FFPACK::IsSingular (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Returns true if the given matrix is singular.

The method is a block elimination with early termination

using LQUP factorization with early termination. If  $M \neq N$ , then the matrix is virtually padded with zeros to make it square and it's determinant is zero.

**Warning**

The input matrix is modified.

**Parameters**

	$F$	base field
	$M$	row dimension of the matrix
	$N$	column dimension of the matrix.
in, out	$A$	input matrix
	$lda$	leading dimension of A

**15.21.3.50 Det()** [1/6]

```
Field::Element& FFPACK::Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P = NULL,
    size_t * Q = NULL )
```

Returns the determinant of the given square matrix.

The method is a block elimination using PLUQ factorization. The input matrix A is overwritten.

**Warning**

The input matrix is modified.

**Parameters**

	<i>F</i>	base field
out	<i>det</i>	the determinant of A
	<i>N</i>	the order of the square matrix A.
in, out	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>psH</i>	(optional) a ParSeqHelper to choose between sequential and parallel execution
	<i>P,Q</i>	(optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification

**15.21.3.51 pDet()**

```
Field::Element& FFPACK::pDet (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0,
    size_t * P = NULL,
    size_t * Q = NULL )
```

**15.21.3.52 Det()** [2/6]

```
Field::Element& FFPACK::Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & pSH,
    size_t * P = NULL,
    size_t * Q = NULL )
```

**15.21.3.53 Solve()** [1/3]

```
Field::Element_ptr FFPACK::Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb )
```

Solves a linear system  $AX = b$  using PLUQ factorization.

@oaram F base field @oaram M matrix order

**Parameters**

in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>x</i>	output solution vector
	<i>incx</i>	increment of x
	<i>b</i>	input right hand side of the system
	<i>incb</i>	increment of b

**15.21.3.54 Solve()** [2/3]

```
Field::Element_ptr FFPACK::Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
```

```
const int incb,
PSHelper & psH )
```

### 15.21.3.55 pSolve()

```
Field::Element_ptr FFPACK::pSolve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb,
    size_t numthreads = 0 )
```

### 15.21.3.56 RandomNullSpaceVector() [1/3]

```
* void FFPACK::RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve  $LX = B$  or  $XL = B$  in place.

$L$  is  $M \times M$  if `Side == FFLAS::FflasLeft` and  $N \times N$  if `Side == FFLAS::FflasRight`,  $B$  is  $M \times N$ . Only the  $R$  non trivial column of  $L$  are stored in the  $M \times R$  matrix  $L$  Requirement : so that  $L$  could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix  $A$ .

#### Parameters

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension $M \times N$ , $A$ is modified to its LU version
	<i>lda</i>	leading dimension of $A$
out	<i>X</i>	output vector
	<i>incX</i>	increment of $X$

**15.21.3.57 NullSpaceBasis()** [1/2]

```

size_t FFPACK::NullSpaceBasis (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & NS,
    size_t & ldn,
    size_t & NSdim )

```

Computes a basis of the Left/Right nullspace of the matrix A.

return the dimension of the nullspace.

**Parameters**

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension M x N, A is modified
	<i>lda</i>	leading dimension of A
out	<i>NS</i>	output matrix of dimension N x NSdim (allocated here)
out	<i>ldn</i>	leading dimension of NS
out	<i>NSdim</i>	the dimension of the Nullspace (N-rank(A))

**15.21.3.58 RowRankProfile()** [1/3]

```

size_t FFPACK::RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Computes the row rank profile of A.

**Parameters**

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix of dimension M x N
	<i>lda</i>	leading dimension of A
out	<i>rkprofile</i>	return the rank profile as an array of row indexes, of dimension r=rank(A)
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

#### Returns

R

#### 15.21.3.59 pRowRankProfile()

```
size_t FFPACK::pRowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive )
```

#### 15.21.3.60 RowRankProfile() [2/3]

```
size_t FFPACK::RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH )
```

#### 15.21.3.61 ColumnRankProfile() [1/3]

```
size_t FFPACK::ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Computes the column rank profile of A.

#### Parameters

	$F$	base field
	$M$	number of rows
Generated by Doxygen		number of columns
in	$A$	input matrix of dimension
	$lda$	leading dimension of A
out	$rkprofile$	return the rank profile as an array of row indexes. of dimension $r=\text{rank}(A)$

A modified rkprofile is allocated during the computation.

#### Returns

R

#### 15.21.3.62 pColumnRankProfile()

```
size_t FFPACK::pColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive )
```

#### 15.21.3.63 ColumnRankProfile() [2/3]

```
size_t FFPACK::ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH )
```

#### 15.21.3.64 RankProfileFromLU()

```
void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag ) [inline]
```

Recovers the column/row rank profile from the permutation of an LU decomposition.

Works with both the CUP/PLE decompositions (obtained by LUdivine) or the PLUQ decomposition. Assumes that the output vector containing the rank profile is already allocated.

## Parameters

	$P$	the permutation carrying the rank profile information
	$N$	the row/col dimension for a row/column rank profile
	$R$	the rank of the matrix
out	$rkprofile$	return the rank profile as an array of indices
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

## 15.21.3.65 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP ) [inline]
```

Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.

Only works with the PLUQ decomposition Assumes that the output vectors containing the rank profiles are already allocated.

## Parameters

$P$	the permutation carrying the rank profile information
$M$	the row dimension of the initial matrix
$N$	the column dimension of the initial matrix
$R$	the rank of the initial matrix
$LSm$	the row dimension of the leading submatrix considered
$LSn$	the column dimension of the leading submatrix considered
$P$	the row permutation of the PLUQ decomposition
$Q$	the column permutation of the PLUQ decomposition
$RRP$	return the row rank profile of the leading submatrix

## Returns

the rank of the  $LSm \times LSn$  leading submatrix

A is modified

## Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.

**15.21.3.66 RowRankProfileSubmatrixIndices()** [1/2]

```

size_t FFPACK::RowRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )

```

RowRankProfileSubmatrixIndices.

Computes the indices of the submatrix  $r \times r$  X of A whose rows correspond to the row rank profile of A.

**Parameters**

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	$R$	list of indices

rowindices and colindices are allocated during the computation. A is modified

**Returns**

R

**15.21.3.67 ColRankProfileSubmatrixIndices()** [1/2]

```

size_t FFPACK::ColRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )

```

Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.

**Parameters**

	$F$	base field
	$M$	number of rows

## Parameters

	$N$	number of columns
in	$A$	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	$R$	list of indices

rowindices and colindices are allocated during the computation.

## Warning

A is modified

## Returns

R

## 15.21.3.68 RowRankProfileSubmatrix() [1/2]

```
size_t FFPACK::RowRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension M x N
	<i>lda</i>	leading dimension of A
out	$X$	the output matrix
out	$R$	list of indices

A is not modified X is allocated during the computation.

## Returns

R

**15.21.3.69 ColRankProfileSubmatrix()** [1/2]

```
size_t FFPACK::ColRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.

**Parameters**

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension M x N
	$lda$	leading dimension of A
out	$X$	the output matrix
out	$R$	list of indices

A is not modified X is allocated during the computation.

**Returns**

R

**15.21.3.70 getTriangular()** [1/2]

```
void FFPACK::getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false )
```

Extracts a triangular matrix from a compact storage A=L\U of rank R.

if OnlyNonZeroVectors is false, then T and A have the same dimensions Otherwise, T is R x N if UpLo = FflasUpper, else T is M x R

## Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	<i>M</i>	row dimension of T
	<i>N</i>	column dimension of T
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored

**Todo** just one triangular fzero+fassign ?

## 15.21.3.71 getTriangular() [2/2]

```
void FFPACK::getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda )
```

Cleans up a compact storage  $A=LU$  to reveal a triangular matrix of rank R.

## Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	<i>diag</i>	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix
in, out	<i>A</i>	input/output matrix
	<i>lda</i>	leading dimension of A

**Todo** just one triangular fzero+fassign ?

**15.21.3.72 getEchelonForm()** [1/2]

```

void FFPACK::getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Extracts a matrix in echelon form from a compact storage  $A=LU$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.

Either  $L$  or  $U$  is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ . row and column dimension of  $T$  are greater or equal to that of  $A$

**Parameters**

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of T
	<i>N</i>	column dimension of T
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

**15.21.3.73 getEchelonForm()** [2/2]

```

void FFPACK::getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,

```

```

const size_t * P,
typename Field::Element_ptr A,
const size_t lda,
const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .

Either  $L$  or  $U$  is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ .

#### Parameters

	$F$	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	$M$	row dimension of $A$
	$N$	column dimension of $A$
	$R$	rank of the triangular matrix (how many rows/columns need to be copied)
	$P$	positions of the $R$ pivots
$in, out$	$A$	input/output matrix
	$lda$	leading dimension of $A$
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

#### 15.21.3.74 getEchelonTransform()

```

void FFPACK::getEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by Row↔ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

#### Parameters

	$F$	base field
	$UpLo$	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form

## Parameters

	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix
	<i>P</i>	permutation matrix
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

15.21.3.75 `getReducedEchelonForm()` [1/2]

```
void FFPACK::getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.

Either  $L$  or  $U$  is in Echelon form (depending on `Uplo`) The echelon structure is defined by the first  $R$  values of the array  $P$ . row and column dimension of  $T$  are greater or equal to that of  $A$

## Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of T
	<i>N</i>	column dimension of T
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored

**15.21.3.76 getReducedEchelonForm()** [2/2]

```

void FFPACK::getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ .

**Parameters**

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
in, out	<i>A</i>	input/output matrix
	<i>lda</i>	leading dimension of A
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

**15.21.3.77 getReducedEchelonTransform()**

```

void FFPACK::getReducedEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Extracts a transformation matrix to echelon form from a compact storage  $A=LU$  of rank  $R$  obtained by Row $\leftrightarrow$ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

#### Parameters

	$F$	base field
	$UpLo$	selects Col (FflasLower) or Row (FflasUpper) Echelon Form
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	$M$	row dimension of $A$
	$N$	column dimension of $A$
	$R$	rank of the triangular matrix
	$P$	permutation matrix
in	$A$	input matrix
	$lda$	leading dimension of $A$
out	$T$	output matrix
	$ldt$	leading dimension of $T$
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

#### 15.21.3.78 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm ) [inline]
```

Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

#### 15.21.3.79 LTBruhatGen()

```
size_t FFPACK::LTBruhatGen (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

LTBruhatGen Suppose  $A$  is Left Triangular Matrix This procedure computes the Bruhat Representation of  $A$  and return the rank of  $A$ .

## Parameters

<i>Fi</i>	base Field
<i>diag</i>	
<i>N</i>	size of A
<i>A</i>	the matrix we search the Bruhat representation
<i>lda</i>	the leading dimension of A
<i>P</i>	a permutation matrix
<i>Q</i>	a permutation matrix

## 15.21.3.80 getLTBruhatGen() [1/2]

```
void FFPACK::getLTBruhatGen (
    const Field & Fi,
    const size_t N,
    const size_t r,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr R,
    const size_t ldr )
```

GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.

## Parameters

<i>Fi</i>	base Field
<i>N</i>	size of the matrix
<i>r</i>	the rank of the matrix
<i>P</i>	a permutation matrix
<i>Q</i>	a permutation matrix
<i>R</i>	the matrix that will contain the rank revealing matrix
<i>ldr</i>	the leading fimension of R

## 15.21.3.81 getLTBruhatGen() [2/2]

```
void FFPACK::getLTBruhatGen (
    const Field & Fi,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    const size_t r,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
```

```

typename Field::Element_ptr T,
const size_t ldt )

```

GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.

#### Parameters

<i>Fi</i>	base Field
<i>Uplo</i>	choose if the procedure return L or U
<i>diag</i>	
<i>N</i>	size of A
<i>r</i>	rank of A
<i>P</i>	permutaion matrix
<i>Q</i>	permutation matrix
<i>A</i>	a bruhat representation
<i>lda</i>	leading dimension of A
<i>T</i>	matrix that will contains L or U
<i>ldt</i>	leading dimension of T

#### 15.21.3.82 LTQSorter()

```

size_t LTQSorter (
    const size_t N,
    const size_t r,
    const size_t * P,
    const size_t * Q ) [inline]

```

LTQSorter This procedure computes the order of quasiseparability of a matrix.

#### Parameters

<i>N</i>	size of the matrix
<i>r</i>	rank of the matrix
<i>P</i>	permutation matrix
<i>Q</i>	permutation matrix

#### 15.21.3.83 CompressToBlockBiDiagonal()

```

size_t FFPACK::CompressToBlockBiDiagonal (
    const Field & Fi,
    const FFLAS::FFLAS_UPLO Uplo,
    size_t N,
    size_t s,
    size_t r,

```

```

    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr A,
    size_t lda,
    typename Field::Element_ptr X,
    size_t ldx,
    size_t * K,
    size_t * M,
    size_t * T )

```

**CompressToBlockBiDiagonal** This procedure compress a compact representation of a row echelon form or column echelon form.

#### Parameters

<i>Fi</i>	base Field
<i>Uplo</i>	chosse if the procedure is based on row or column
<i>N</i>	size of the matrix
<i>s</i>	order of qausiseparability
<i>r</i>	rank
<i>P</i>	permutation matrix
<i>Q</i>	permutation matrix
<i>A</i>	the matrix to compact
<i>lda</i>	leading dimension of A
<i>X</i>	matrix that will stock the representation
<i>ldx</i>	leading dimension of X
<i>K</i>	stock the position of the blocks in A
<i>M</i>	permutation matrix
<i>T</i>	stock the operation done in the procedure

#### 15.21.3.84 ExpandBlockBiDiagonalToBruhat()

```

void FFPACK::ExpandBlockBiDiagonalToBruhat (
    const Field & Fi,
    const FFLAS::FFLAS_UPLO Uplo,
    size_t N,
    size_t s,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda,
    typename Field::Element_ptr X,
    size_t ldx,
    size_t NbBlocks,
    size_t * K,
    size_t * M,
    size_t * T )

```

**ExpandBlockBiDiagonal** This procedure expand a compact representation of a row echelon form or column echelon form.

## Parameters

<i>Fi</i>	base Field
<i>Uplo</i>	chosse if the procedure is based on row or column
<i>N</i>	size of the matrix
<i>s</i>	order of qausiseparability
<i>r</i>	rank
<i>A</i>	the matrix that will sotck the expanded representation
<i>lda</i>	leading dimension of A
<i>X</i>	matrix to expand
<i>ldx</i>	leading dimension of X
<i>K</i>	stock the position of the blocks in A
<i>M</i>	permutation matrix
<i>T</i>	stock the operation done in the procedure

**15.21.3.85 Bruhat2EchelonPermutation()**

```
void Bruhat2EchelonPermutation (
    size_t N,
    size_t R,
    const size_t * P,
    const size_t * Q,
    size_t * M ) [inline]
```

Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.

## Parameters

in	<i>N</i>	size of the matrix
in	<i>R</i>	rank
in	<i>P</i>	permutation Matrix
in	<i>Q</i>	permutation Matrix
out	<i>M</i>	output permutation matrix

**15.21.3.86 Tinverter() [1/2]**

```
size_t* FFPACK::Tinverter (
    size_t * T,
    size_t r )
```

**15.21.3.87 ComputeRPermutation()** [1/2]

```
void FFPACK::ComputeRPermutation (
    const Field & Fi,
    size_t N,
    size_t r,
    const size_t * P,
    const size_t * Q,
    size_t * R,
    size_t * MU,
    size_t * ML )
```

**15.21.3.88 productBruhatxTS()** [1/2]

```
void FFPACK::productBruhatxTS (
    const Field & Fi,
    size_t N,
    size_t s,
    size_t r,
    const size_t * P,
    const size_t * Q,
    const typename Field::Element_ptr Xu,
    size_t ldu,
    size_t NbBlocksU,
    size_t * Ku,
    size_t * Tu,
    size_t * MU,
    const typename Field::Element_ptr Xl,
    size_t ldl,
    size_t NbBlocksL,
    size_t * Kl,
    size_t * Tl,
    size_t * ML,
    typename Field::Element_ptr B,
    size_t t,
    size_t ldb,
    typename Field::Element_ptr C,
    size_t ldc )
```

productBruhatxTS Compute the product between the CRE compact representation of a matrix A and B a tall matrix

**15.21.3.89 LQUPtoInverseOfFullRankMinor()** [1/2]

```
Field::Element_ptr FFPACK::LQUPtoInverseOfFullRankMinor (
    const Field & F,
    const size_t rank,
    typename Field::Element_ptr A_factors,
    const size_t lda,
    const size_t * QtPointer,
    typename Field::Element_ptr X,
    const size_t ldx )
```

LQUPtoInverseOfFullRankMinor.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal r x r submatrix This procedure efficiently computes the inverse of this minor and puts it into X.

**Note**

It changes the lower entries of `A_factors` in the process (NB: unless `A` was nonsingular and square)

**Parameters**

<i>F</i>	base field
<i>rank</i>	rank of the matrix.
<i>A_factors</i>	matrix containing the L and U entries of the factorization
<i>lda</i>	leading dimension of A
<i>QtPointer</i>	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
<i>X</i>	desired location for output
<i>ldx</i>	leading dimension of X

**15.21.3.90 RandomNullSpaceVector() [2/3]**

```
void FFPACK::RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve  $LX = B$  or  $XL = B$  in place.

$L$  is  $M \times M$  if `Side == FFLAS::FflasLeft` and  $N \times N$  if `Side == FFLAS::FflasRight`,  $B$  is  $M \times N$ . Only the  $R$  non trivial column of  $L$  are stored in the  $M \times R$  matrix  $L$  Requirement : so that  $L$  could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix  $A$ .

**Parameters**

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
<i>in, out</i>	<i>A</i>	input matrix of dimension $M \times N$ , $A$ is modified to its LU version
	<i>lda</i>	leading dimension of A
<i>out</i>	<i>X</i>	output vector
	<i>incX</i>	increment of X

**15.21.3.91 solveLB() [1/2]**

```
void FFPACK::solveLB (
    const Field & F,
```

```

    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )

```

### 15.21.3.92 solveLB2() [1/2]

```

void FFPACK::solveLB2 (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )

```

### 15.21.3.93 TInverter() [2/2]

```

size_t* FFPACK::TInverter (
    const size_t * T,
    size_t r ) [inline]

```

### 15.21.3.94 ComputeRPermutation() [2/2]

```

void FFPACK::ComputeRPermutation (
    const Field & Fi,
    size_t N,
    size_t r,
    const size_t * P,
    const size_t * Q,
    size_t * R,
    const size_t * MU,
    const size_t * ML ) [inline]

```

### 15.21.3.95 expandLCRE()

```
Field::Element_ptr FFPACK::expandLCRE (
    const Field & Fi,
    size_t N,
    size_t s,
    size_t r,
    size_t * R,
    size_t i,
    typename Field::ConstElement_ptr Xu,
    size_t ldu,
    size_t NbBlocksU,
    const size_t * Ku,
    const size_t * Tuinv,
    typename Field::ConstElement_ptr Xl,
    size_t ldl,
    size_t NbBlocksL,
    const size_t * Kl,
    const size_t * Tlinv,
    typename Field::Element_ptr CRE,
    size_t ldcre ) [inline]
```

Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.

### 15.21.3.96 productBruhatxTS() [2/2]

```
void FFPACK::productBruhatxTS (
    const Field & Fi,
    size_t N,
    size_t s,
    size_t r,
    size_t t,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr Xu,
    size_t ldu,
    size_t NbBlocksU,
    const size_t * Ku,
    const size_t * Tu,
    const size_t * MU,
    typename Field::ConstElement_ptr Xl,
    size_t ldl,
    size_t NbBlocksL,
    const size_t * Kl,
    const size_t * Tl,
    const size_t * ML,
    typename Field::Element_ptr B,
    size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr D,
    size_t ldd ) [inline]
```

Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B:  $C \leftarrow A \times B + \text{beta}C$ .

## Parameters

	$F$	the base field
	$N$	the order of $A$
	$s$	the order of quasiseparability of $A$
	$r$	the number of pivots in the left-triangular part of the rank profile matrix of $A$
	$t$	the number of columns of $B$
	$P$	the row indices of the pivots of $A$
	$Q$	the column indices of the pivots of $A$
	$X_u$	the compact storage of $U$ : $D_u$ blocks in the first $s$ rows, $S_u$ blocks in the last $s$ rows
	$ldxu$	the leading dimension of $X_u$
	$NbBlocksU$	the number of diagonal blocks in the compact storage of $U$
	$K_u$	the list of starting column positions for each block of the storage of $U$
	$T_u$	the folding matrix for the compact storage of $U$ : $D_u + T_u S_u$ is in row echelon form
	$M_u$	a permutation matrix such that $M_u(D_u + T_u S_u)$ is the $U$ factor of the Bruhat generator
	$X_l$	the compact storage of $L$ : $D_l$ blocks in the first $s$ columns, $S_l$ blocks in the last $s$ columns
	$ldxl$	the leading dimension of $X_l$
	$NbBlocksL$	the number of diagonal blocks in the compact storage of $L$
	$K_l$	the list of starting row positions for each block of the storage of $L$
	$T_l$	the folding matrix for the compact storage of $L$ : $D_l + S_l T_l$ is in column echelon form
	$M_l$	a permutation matrix such that $(D_l + S_l T_l) M_l$ is the $L$ factor of the Bruhat generator
	$B$	an $N \times t$ dense matrix
	$ldb$	leading dimension of $B$
	$\beta$	scaling constant
in, out	$C$	output matrix
	$ldc$	leading dimension of $C$

**Bibliography** Pernet C. and Storjohann A. *Time and space efficient generators for quasiseparable matrices*, JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010

## 15.21.3.97 Danilevski()

```
std::list<Polynomial>& FFPACK::Danilevski (
    const Field & F,
    std::list<Polynomial> & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

**15.21.3.98 buildMatrix()**

```
Field::Element_ptr FFPACK::buildMatrix (
    const Field & F,
    typename Field::ConstElement_ptr E,
    typename Field::ConstElement_ptr C,
    const size_t lda,
    const size_t * B,
    const size_t * T,
    const size_t me,
    const size_t mc,
    const size_t lambda,
    const size_t mu )
```

**Bug** is this :

**15.21.3.99 CharPoly() [4/8]**

```
FFPACK::RNSInteger<FFPACK::rns_double>::Element_ptr FFPACK::CharPoly (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.21.3.100 CharPoly() [5/8]**

```
Givaro::Poly1Dom<Givaro::ZRing<Givaro::Integer> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > & R,
    Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & charp,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.21.3.101 Det() [3/6]**

```
FFPACK::RNSInteger<FFPACK::rns_double>::Element_ptr& FFPACK::Det (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & det,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    const PSHelper & psH ) [inline]
```

**15.21.3.102 Det()** [4/6]

```
Givaro::Integer& FFPACK::Det (
    const Givaro::ZRing< Givaro::Integer > & F,
    Givaro::Integer & det,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    const PSHelper & pSH,
    size_t * P,
    size_t * Q ) [inline]
```

**15.21.3.103 fsytrf\_BC\_Crout()**

```
bool FFPACK::fsytrf_BC_Crout (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

**15.21.3.104 fsytrf\_BC\_RL()**

```
size_t FFPACK::fsytrf_BC_RL (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

**15.21.3.105 fsytrf\_UP\_RPM\_BC\_RL()**

```
size_t FFPACK::fsytrf_UP_RPM_BC_RL (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.21.3.106 fsytrf\_LOW\_RPM\_BC\_Crout()**

```
size_t FFPACK::fsytrf_LOW_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.21.3.107 fsytrf\_UP\_RPM\_BC\_Crout()**

```
size_t FFPACK::fsytrf_UP_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.21.3.108 fsytrf\_UP\_RPM()**

```
size_t FFPACK::fsytrf_UP_RPM (
    const Field & Fi,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P,
    size_t BCThreshold ) [inline]
```

MathP <- [ [ I ] x P1 ] [ [ I (N1+R2) ] [ P2^T ] ] x [ P3^T ] [ ----- | --- ] [ [ Q2^T ]

Changing [ U1 V1 | E1 E21 E22 ] into [ U1 E11 E12 V1 E\* E\* ] [ 0 | L2 \ U2 V21 V22 ] [ U4 V41 0 V42 V43 ] [ 0 | M2 0 0 ] [ U3 0 0 V3 ] [ ----- | ----- ] [ [ 0 0 0 ] [ 0 | H1 H21 H22 ] [ 0 | U3 V3 ] [ 0 | 0 ] where U4 is the 2R2 x 2R2 matrix formed by interleaving U2, L2^T and H1

**15.21.3.109 fsytrf\_nonunit() [2/3]**

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Sequential seq,
    size_t threshold ) [inline]
```

**15.21.3.110 fsytrf\_nonunit()** [3/3]

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    size_t threshold ) [inline]
```

**15.21.3.111 fsytrf\_RPM()**

```
size_t FFPACK::fsytrf_RPM (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t threshold ) [inline]
```

**15.21.3.112 getTridiagonal()**

```
void FFPACK::getTridiagonal (
    const Field & F,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    size_t * P,
    typename Field::Element_ptr T,
    const size_t ldt ) [inline]
```

**15.21.3.113 LUdivine\_gauss()** [1/2]

```
size_t FFPACK::LUdivine_gauss (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

**15.21.3.114 LUdivine\_small() [1/2]**

```

size_t FFPACK::LUdivine_small (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

**15.21.3.115 LUdivine() [2/4]**

```

size_t FFPACK::LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

**Todo** std::swap ?

**15.21.3.116 LUdivine() [3/4]**

```

size_t FFPACK::LUdivine (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

**15.21.3.117 MonotonicCompress()**

```
void FFPACK::MonotonicCompress (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv ) [inline]
```

**15.21.3.118 MonotonicCompressMorePivots()**

```
void FFPACK::MonotonicCompressMorePivots (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t rowstomove,
    const size_t lenP ) [inline]
```

**15.21.3.119 MonotonicCompressCycles()**

```
void FFPACK::MonotonicCompressCycles (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t lenP ) [inline]
```

**15.21.3.120 MonotonicExpand()**

```

void FFPACK::MonotonicExpand (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv )

```

**15.21.3.121 applyP\_block()**

```

void FFPACK::applyP_block (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]

```

**15.21.3.122 doApplyS()**

```

void FFPACK::doApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.21.3.123 MatrixApplyS() [1/3]**

```
void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.21.3.124 MatrixApplyS() [2/3]**

```
void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

**15.21.3.125 MatrixApplyS() [3/3]**

```
void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.21.3.126 PermApplyS()**

```
void FFPACK::PermApplyS (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.21.3.127 doApplyT()**

```
void FFPACK::doApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.21.3.128 MatrixApplyT() [1/3]**

```
void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.21.3.129 MatrixApplyT() [2/3]**

```
void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
```

```

const size_t R1,
const size_t R2,
const size_t R3,
const size_t R4,
const FFLAS::ParSeqHelper::Sequential seq ) [inline]

```

### 15.21.3.130 MatrixApplyT() [3/3]

```

void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

### 15.21.3.131 PermApplyT()

```

void FFPACK::PermApplyT (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

### 15.21.3.132 composePermutationsLLL()

```

void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]

```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

#### Parameters

in, out	$P1$	a LAPACK permutation of size N
	$P2$	a LAPACK permutation of size N-R

**15.21.3.133 composePermutationsLLM()**

```
void composePermutationsLLM (
    size_t * MathP,
    const size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.

**Parameters**

out		
-----	--	--

**15.21.3.134 composePermutationsMLM()**

```
void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.

**Parameters**

in, out	<i>MathP1</i>	a $\text{MathPermutation}$ of size N
	<i>P2</i>	a LAPACK permutation of size N-R

**15.21.3.135 cyclic\_shift\_mathPerm()**

```
void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s ) [inline]
```

**15.21.3.136 cyclic\_shift\_row\_col() [1/2]**

```
void FFPACK::cyclic_shift_row_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.21.3.137 cyclic\_shift\_row() [1/3]**

```
void FFPACK::cyclic_shift_row (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.21.3.138 cyclic\_shift\_row() [2/3]**

```
void FFPACK::cyclic_shift_row (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.21.3.139 cyclic\_shift\_col() [1/3]**

```
void FFPACK::cyclic_shift_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.21.3.140 cyclic\_shift\_col() [2/3]**

```
void FFPACK::cyclic_shift_col (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.21.3.141 PLUQ\_basecaseV3()**

```
size_t FFPACK::PLUQ_basecaseV3 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.21.3.142 PLUQ\_basecaseV2()**

```
size_t FFPACK::PLUQ_basecaseV2 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.21.3.143 PLUQ\_basecaseCrout()**

```
size_t FFPACK::PLUQ_basecaseCrout (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.21.3.144 \_PLUQ()**

```
size_t FFPACK::_PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold ) [inline]
```

**15.21.3.145 PLUQ()** [4/6]

```

size_t FFPACK::PLUQ (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper ) [inline]

```

**15.21.3.146 threads\_fgemm()**

```

void FFPACK::threads_fgemm (
    const size_t m,
    const size_t n,
    const size_t r,
    int nbthreads,
    size_t * W1,
    size_t * W2,
    size_t * W3,
    size_t gamma )

```

**15.21.3.147 threads\_ftrsm()**

```

void FFPACK::threads_ftrsm (
    const size_t m,
    const size_t n,
    int nbthreads,
    size_t * t1,
    size_t * t2 )

```

**15.21.3.148 PLUQ()** [5/6]

```

size_t FFPACK::PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & PSHelper ) [inline]

```

**15.21.3.149 fflas\_const\_cast()** [1/3]

```
rns_double_elt_ptr FFPACK::fflas_const_cast (
    rns_double_elt_cstptr x ) [inline]
```

**15.21.3.150 fflas\_const\_cast()** [2/3]

```
rns_double_elt_cstptr FFPACK::fflas_const_cast (
    rns_double_elt_ptr x ) [inline]
```

**15.21.3.151 cyclic\_shift\_row\_col()** [2/2]

```
void FFPACK::cyclic_shift_row_col (
    Base_t * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.21.3.152 cyclic\_shift\_row()** [3/3]

```
template INST_OR_DECL void FFPACK::cyclic_shift_row (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.21.3.153 cyclic\_shift\_col()** [3/3]

```
template INST_OR_DECL void FFPACK::cyclic_shift_col (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.21.3.154 applyP() [4/4]**

```
template INST_OR_DECL void FFPACK::applyP (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P )
```

**15.21.3.155 fgetrs() [3/4]**

```
template INST_OR_DECL void FFPACK::fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.21.3.156 fgetrs() [4/4]**

```
template INST_OR_DECL FFLAS_ELT* FFPACK::fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * X,
    const size_t ldX,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.21.3.157 fgesv() [3/4]**

```
template INST_OR_DECL size_t FFPACK::fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.21.3.158 fgesv() [4/4]**

```
template INST_OR_DECL size_t FFPACK::fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldX,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.21.3.159 ftrtri() [2/2]**

```
template INST_OR_DECL void FFPACK::ftrtri (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t threshold )
```

**15.21.3.160 trinv\_left() [2/2]**

```
template INST_OR_DECL void FFPACK::trinv_left (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * L,
    const size_t ldL,
    FFLAS_ELT * X,
    const size_t ldX )
```

**15.21.3.161 ftrtrm()** [2/2]

```
template INST_OR_DECL void FFPACK::ftrtrm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.21.3.162 PLUQ()** [6/6]

```
template INST_OR_DECL size_t FFPACK::PLUQ (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.21.3.163 LUdivine()** [4/4]

```
template INST_OR_DECL size_t FFPACK::LUdivine (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag,
    const size_t cutoff )
```

**15.21.3.164 LUdivine\_small()** [2/2]

```
template INST_OR_DECL size_t FFPACK::LUdivine_small (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.165 LUdivine\_gauss() [2/2]**

```
template INST_OR_DECL size_t FFPACK::LUdivine_gauss (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.166 RowEchelonForm() [3/3]**

```
template INST_OR_DECL size_t FFPACK::RowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.167 ReducedRowEchelonForm() [3/3]**

```
template INST_OR_DECL size_t FFPACK::ReducedRowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.168 ColumnEchelonForm() [3/3]**

```
template INST_OR_DECL size_t FFPACK::ColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.169 ReducedColumnEchelonForm()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ReducedColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.170 Invert()** [3/4]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    int & nullity )
```

**15.21.3.171 Invert()** [4/4]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

**15.21.3.172 Invert2()** [2/2]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

**15.21.3.173 CharPoly() [6/8]**

```
template INST_OR_DECL std::list<Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element>& FFPACK←
::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.21.3.174 CharPoly() [7/8]**

```
template INST_OR_DECL Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.21.3.175 CharPoly() [8/8]**

```
template INST_OR_DECL Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.21.3.176 MinPoly() [3/4]**

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G )
```

**15.21.3.177 MinPoly()** [4/4]

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda )
```

**15.21.3.178 MatVecMinPoly()** [2/2]

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MatVecMinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * V,
    const size_t incv )
```

**15.21.3.179 KrylovElim()**

```
template INST_OR_DECL size_t FFPACK::KrylovElim (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const size_t deg,
    size_t * iterates,
    size_t * inviterates,
    const size_t maxit,
    size_t virt )
```

**15.21.3.180 SpecRankProfile()**

```
template INST_OR_DECL size_t FFPACK::SpecRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t deg,
    size_t * rankProfile )
```

**15.21.3.181 Rank()** [3/3]

```
template INST_OR_DECL size_t FFPACK::Rank (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.21.3.182 IsSingular()** [2/2]

```
template INST_OR_DECL bool FFPACK::IsSingular (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.21.3.183 Det()** [5/6]

```
template INST_OR_DECL FFLAS_ELT& FFPACK::Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.21.3.184 Det()** [6/6]

```
template INST_OR_DECL FFLAS_ELT& FFPACK::Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & parH,
    size_t * P,
    size_t * Q )
```

**15.21.3.185 Solve()** [3/3]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Solve (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * x,
    const int incx,
    const FFLAS_ELT * b,
    const int incb )
```

**15.21.3.186 solveLB()** [2/2]

```
template INST_OR_DECL void FFPACK::solveLB (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.21.3.187 solveLB2()** [2/2]

```
template INST_OR_DECL void FFPACK::solveLB2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.21.3.188 RandomNullSpaceVector()** [3/3]

```
template INST_OR_DECL void FFPACK::RandomNullSpaceVector (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t incX )
```

**15.21.3.189 NullSpaceBasis()** [2/2]

```
template INST_OR_DECL size_t FFPACK::NullSpaceBasis (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& NS,
    size_t & ldn,
    size_t & NSdim )
```

**15.21.3.190 RowRankProfile()** [3/3]

```
template INST_OR_DECL size_t FFPACK::RowRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.191 ColumnRankProfile()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ColumnRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.192 RowRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t FFPACK::RowRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.21.3.193 ColRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t FFPACK::ColRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.21.3.194 RowRankProfileSubmatrix()** [2/2]

```
template INST_OR_DECL size_t FFPACK::RowRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.21.3.195 ColRankProfileSubmatrix()** [2/2]

```
template INST_OR_DECL size_t FFPACK::ColRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.21.3.196 getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >()** [1/2]

```
template INST_OR_DECL void FFPACK::getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors )
```

**15.21.3.197** `getTriangular< FFLAS_FIELD< FFLAS_ELT > >()` [2/2]

```
template INST_OR_DECL void FFPACK::getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.21.3.198** `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()` [1/2]

```
template INST_OR_DECL void FFPACK::getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.199** `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()` [2/2]

```
template INST_OR_DECL void FFPACK::getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.200 getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()**

```
template INST_OR_DECL void FFPACK::getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.201 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [1/2]**

```
template INST_OR_DECL void FFPACK::getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.202 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [2/2]**

```
template INST_OR_DECL void FFPACK::getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.203 getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()**

```
template INST_OR_DECL void FFPACK::getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )
```

**15.21.3.204 LQUPToInverseOfFullRankMinor() [2/2]**

```
template INST_OR_DECL FFLAS_ELT* FFPACK::LQUPToInverseOfFullRankMinor (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t rank,
    FFLAS_ELT * A_factors,
    const size_t lda,
    const size_t * QtPointer,
    FFLAS_ELT * X,
    const size_t ldx )
```

**15.21.3.205 fflas\_const\_cast() [3/3]**

```
T FFPACK::fflas_const_cast (
    CT x )
```

**15.21.3.206 failure()**

```
Failure& FFPACK::failure ( ) [inline]
```

**15.21.3.207 isOdd() [1/3]**

```
bool FFPACK::isOdd (
    const T & a ) [inline]
```

**15.21.3.208 isOdd()** [2/3]

```
bool FFPACK::isOdd (
    const float & a ) [inline]
```

**15.21.3.209 isOdd()** [3/3]

```
bool FFPACK::isOdd (
    const double & a ) [inline]
```

**15.21.3.210 NonZeroRandomMatrix()** [1/2]

```
Field::Element_ptr FFPACK::NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

**Parameters**

	$F$	field
	$m$	number of rows in A
	$n$	number of cols in A
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of A
	$G$	a random iterator

**Returns**

A.

**15.21.3.211 NonZeroRandomMatrix()** [2/2]

```
Field::Element_ptr FFPACK::NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
```

```

typename Field::Element_ptr A,
size_t lda ) [inline]

```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

#### Parameters

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

#### Returns

$A$ .

#### 15.21.3.212 RandomMatrix() [1/2]

```

Field::Element_ptr FFPACK::RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]

```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

#### Parameters

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$
	$G$	a random iterator

#### Returns

$A$ .

**15.21.3.213 RandomMatrix()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

**Parameters**

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

**Returns**

$A$ .

**15.21.3.214 RandomTriangularMatrix()** [1/2]

```
Field::Element_ptr FFPACK::RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

**Parameters**

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
	$UpLo$	whether $A$ is upper or lower triangular
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$
	$G$	a random iterator

## Returns

A.

### 15.21.3.215 RandomTriangularMatrix() [2/2]

```
Field::Element_ptr FFPACK::RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

## Parameters

	$F$	field
	$m$	number of rows in A
	$n$	number of cols in A
	$UpLo$	whether A is upper or lower triangular
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of A

## Returns

A.

### 15.21.3.216 RandInt()

```
size_t FFPACK::RandInt (
    size_t a,
    size_t b ) [inline]
```

**15.21.3.217 RandomSymmetricMatrix()**

```
Field::Element_ptr FFPACK::RandomSymmetricMatrix (
    const Field & F,
    size_t n,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

**Parameters**

	$F$	field
	$n$	order of $A$
out	$A$	the matrix (preallocated to at least $n \times lda$ field elements)
	$lda$	leading dimension of $A$
	$G$	a random iterator

**Returns**

$A$ .

**15.21.3.218 RandomMatrixWithRank() [1/2]**

```
Field::Element_ptr FFPACK::RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$m$	number of rows in $A$
$n$	number of cols in $A$
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of $A$
$G$	a random iterator

**Returns**

$A$ .

**15.21.3.219 RandomMatrixWithRank()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
	$r$	rank of the matrix to build
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

**Returns**

$A$ .

**15.21.3.220 RandomIndexSubset()**

```
size_t* FFPACK::RandomIndexSubset (
    size_t N,
    size_t R,
    size_t * P ) [inline]
```

Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.

**Parameters**

	$N$	the cardinality of the sampling set
	$R$	the number of elements to sample
out	$P$	the output sequence (pre-allocated to at least $R$ indices)

**15.21.3.221 RandomPermutation()**

```
size_t* FFPACK::RandomPermutation (
    size_t N,
    size_t * P ) [inline]
```

Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.

**Parameters**

	$N$	the length of the permutation
out	$P$	the output permutation (pre-allocated to at least $N$ indices)

**15.21.3.222 RandomRankProfileMatrix()**

```
void FFPACK::RandomRankProfileMatrix (
    size_t M,
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.

**Parameters**

	$M$	row dimension
	$N$	column dimension
out	$rows$	the row position of each non zero element (pre-allocated)
out	$cols$	the column position of each non zero element (pre-allocated)

**15.21.3.223 swapval()**

```
void FFPACK::swapval (
    size_t k,
    size_t N,
    size_t * P,
    size_t val ) [inline]
```

**15.21.3.224 RandomSymmetricRankProfileMatrix()**

```
void FFPACK::RandomSymmetricRankProfileMatrix (
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random a symmetric R-subpermutation of dimension  $N \times N$  : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

**Parameters**

	$N$	matrix order
out	<i>rows</i>	the row position of each non zero element (pre-allocated)
out	<i>cols</i>	the column position of each non zero element (pre-allocated)

**15.21.3.225 RandomLTQSRankProfileMatrix()**

```
void FFPACK::RandomLTQSRankProfileMatrix (
    size_t n,
    size_t r,
    size_t t,
    size_t * rows,
    size_t * cols ) [inline]
```

**15.21.3.226 RandomMatrixWithRankandRPM() [1/2]**

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$m$	number of rows in A
$n$	number of cols in A
$r$	rank of the matrix to build

## Parameters

<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
<i>lda</i>	leading dimension of <i>A</i>
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

## Returns

*A*.

## 15.21.3.227 RandomMatrixWithRankandRPM() [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

## Parameters

<i>F</i>	field
<i>m</i>	number of rows in <i>A</i>
<i>n</i>	number of cols in <i>A</i>
<i>r</i>	rank of the matrix to build
<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
<i>lda</i>	leading dimension of <i>A</i>
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots

## Returns

*A*.

## 15.21.3.228 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
```

```

    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]

```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

#### Parameters

<i>F</i>	field
<i>n</i>	order of A
<i>r</i>	rank of A
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

#### Returns

A.

### 15.21.3.229 RandomSymmetricMatrixWithRankandRPM() [2/2]

```

Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]

```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

#### Parameters

<i>F</i>	field
<i>n</i>	order of A
<i>r</i>	rank of A
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots

## Returns

A.

**15.21.3.230 RandomMatrixWithRankandRandomRPM()** [1/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

## Parameters

$F$	field
$m$	number of rows in A
$n$	number of cols in A
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of A
$G$	a random iterator

## Returns

A.

**15.21.3.231 RandomMatrixWithRankandRandomRPM()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

## Parameters

$F$	field
-----	-------

## Parameters

<i>m</i>	number of rows in A
<i>n</i>	number of cols in A
<i>r</i>	rank of the matrix to build
<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
<i>lda</i>	leading dimension of A

## Returns

A.

### 15.21.3.232 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

## Parameters

<i>F</i>	field
<i>n</i>	order of A
<i>r</i>	rank of A
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>G</i>	a random iterator

## Returns

A.

### 15.21.3.233 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
```

```
typename Field::Element_ptr A,  
size_t lda ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

## Parameters

<i>F</i>	field
<i>n</i>	order of $A$
<i>r</i>	rank of $A$
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of $A$

## Returns

$A$ .

## 15.21.3.234 RandomMatrixWithDet() [1/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

## Parameters

<i>F</i>	field
<i>d</i>	the prescribed value for the determinant of $A$
<i>n</i>	number of cols in $A$
<i>A</i>	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of $A$

## Returns

$A$ .

## 15.21.3.235 RandomMatrixWithDet() [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
```

```
size_t lda,  
RandIter & G ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

## Parameters

$F$	field
$d$	the prescribed value for the determinant of $A$
$n$	number of cols in $A$
$A$	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
$lda$	leading dimension of $A$

## Returns

$A$ .

## 15.21.3.236 RandomLTQSMATRIXWithRankandQSORDER()

```
Field::Element_ptr FFPACK::RandomLTQSMATRIXWithRankandQSORDER (
    Field & F,
    size_t n,
    size_t r,
    size_t t,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

## 15.21.3.237 chooseField()

```
Field* FFPACK::chooseField (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

## 15.21.3.238 chooseField&lt; Givaro::ZRing&lt; int32\_t &gt; &gt;()

```
Givaro::ZRing<int32_t>* FFPACK::chooseField< Givaro::ZRing< int32_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

## 15.21.3.239 chooseField&lt; Givaro::ZRing&lt; int64\_t &gt; &gt;()

```
Givaro::ZRing<int64_t>* FFPACK::chooseField< Givaro::ZRing< int64_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.21.3.240 chooseField< Givaro::ZRing< float > >()**

```
Givaro::ZRing<float>* FFPACK::chooseField< Givaro::ZRing< float > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.21.3.241 chooseField< Givaro::ZRing< double > >()**

```
Givaro::ZRing<double>* FFPACK::chooseField< Givaro::ZRing< double > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.22 FFPACK::Protected Namespace Reference****Functions**

- template<class Field >  
size\_t LUdivine\_construct (const Field &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, typename Field::Element\_ptr u, const size\_t incu, size\_t \*P, bool computeX, const FFPACK\_MINPOLY\_TAG MinTag=FFpackDense, const size\_t kg\_mc=0, const size\_t kg\_mb=0, const size\_t kg\_j=0)
- template<class Field >  
size\_t GaussJordan (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag)  
*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*
- template<class Field , class Polynomial >  
std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda)
- template<class Field , class Polynomial >  
int KGFast (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*kg\_mc, size\_t \*kg\_mb, size\_t \*kg\_j)
- template<class Field , class Polynomial >  
std::list< Polynomial > & KGFast\_generalized (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)
- template<class Field >  
void fgmv\_kgf (const Field &F, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)
- template<class Field , class Polynomial , class RandIter >  
std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr U, const size\_t ldu, RandIter &G)
- template<class Field , class Polynomial >  
std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)
- template<class PolRing >  
void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, size\_t &Nb, typename PolRing::Domain\_t::Element\_ptr &B, size\_t &ldb, typename PolRing::Domain\_t::RandIter &g, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)

- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::↵`  
`Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda,`  
`const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N,`  
`typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const`  
`size_t ldk, size_t *P)`
- `template<class Field , class Polynomial >`  
`Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P,`  
`const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_↵`  
`mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t updatedD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > >`  
`&minpt)`
- `template<class Field >`  
`size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename`  
`Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`  
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_↵`  
`_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const`  
`size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const`  
`size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const`  
`size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_↵`  
`_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t`  
`ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK↵`  
`_MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

## 15.22.1 Function Documentation

## 15.22.1.1 LUdivine\_construct() [1/2]

```

size_t FFPACK::Protected::LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK_MINPOLY_TAG MinTag = FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )

```

## 15.22.1.2 GaussJordan()

```

size_t GaussJordan (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.

- Bibliography**
- Algorithm 2.8 of A. Storjohann Thesis 2000,
  - Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013

## Parameters

	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
in, out	<i>A</i>	an m x n matrix
	<i>lda</i>	leading dimension of A
	<i>P</i>	row permutation
	<i>Q</i>	column permutation
	<i>LuTag</i>	set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon



```

typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr U,
const size_t ldu,
RandIter & G )

```

### 15.22.1.8 Danilevski()

```

std::list<Polynomial>& FFPACK::Protected::Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

### 15.22.1.9 RandomKrylovPrecond()

```

void RandomKrylovPrecond (
    const PolRing & PR,
    std::list< typename PolRing::Element > & completedFactors,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    size_t & Nb,
    typename PolRing::Domain_t::Element_ptr & B,
    size_t & ldb,
    typename PolRing::Domain_t::RandIter & g,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

**Todo** swap to save space ??

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

**Todo** swap to save space ??

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

### 15.22.1.10 ArithProg()

```

std::list< typename PolRing::Element > & ArithProg (
    const PolRing & PR,
    std::list< typename PolRing::Element > & frobeniusForm,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const size_t degree ) [inline]

```

### 15.22.1.11 LUKrylov\_KGFast()

```

std::list<Polynomial>& FFPACK::Protected::LUKrylov_KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,

```

```

typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr X,
const size_t ldX )

```

#### 15.22.1.12 MatVecMinPoly()

```

Polynomial& FFPACK::Protected::MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr v,
    const size_t incv,
    typename Field::Element_ptr K,
    const size_t ldK,
    size_t * P )

```

#### 15.22.1.13 Hybrid\_KGF\_LUK\_MinPoly()

```

Polynomial& FFPACK::Protected::Hybrid_KGF_LUK_MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    size_t * P,
    const FFPACK_MINPOLY_TAG MinTag = FFPACK::FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )

```

#### 15.22.1.14 updateD()

```

size_t FFPACK::Protected::updateD (
    const Field & F,
    size_t * d,
    size_t k,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

#### 15.22.1.15 newD()

```

size_t FFPACK::Protected::newD (
    const Field & F,
    size_t * d,
    bool & KeepOn,
    const size_t l,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t * Q,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

#### 15.22.1.16 CompressRows()

```
void FFPACK::Protected::CompressRows (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

#### 15.22.1.17 CompressRowsQK()

```
void FFPACK::Protected::CompressRowsQK (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

#### 15.22.1.18 DeCompressRows()

```
void FFPACK::Protected::DeCompressRows (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

#### 15.22.1.19 DeCompressRowsQK()

```
void FFPACK::Protected::DeCompressRowsQK (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

#### 15.22.1.20 CompressRowsQA()

```
void FFPACK::Protected::CompressRowsQA (
    Field & F,
```

```

const size_t M,
typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr tmp,
const size_t ldtmp,
const size_t * d,
const size_t nb_blocs ) [inline]

```

#### 15.22.1.21 DeCompressRowsQA()

```

void FFPACK::Protected::DeCompressRowsQA (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

#### 15.22.1.22 LUdivine\_construct() [2/2]

```

size_t FFPACK::Protected::LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK::FFPACK_MINPOLY_TAG MinTag,
    const size_t kg_mc,
    const size_t kg_mb,
    const size_t kg_j )

```

## 15.23 Givaro Namespace Reference

### Data Structures

- class [ModularBalanced](#)
- class [Montgomery](#)

## 15.24 MKL\_CONFIG Namespace Reference

## 15.25 Reclnt Namespace Reference

### Data Structures

- class [rint](#)

- class [ruint](#)



## Chapter 16

# Data Structure Documentation

### 16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Winograd value](#)

#### 16.1.1 Member Typedef Documentation

##### 16.1.1.1 value

typedef [MMHelperAlgo::Winograd value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Classic value](#)

#### 16.2.1 Member Typedef Documentation

##### 16.2.1.1 value

typedef [MMHelperAlgo::Classic value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.3 ALL< v > Struct Template Reference

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.4 ALL< false, v... > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

#### 16.4.1 Field Documentation

##### 16.4.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.5 ALL< true, v... > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = ALL<v...>::value

#### 16.5.1 Field Documentation

##### 16.5.1.1 value

```
constexpr bool value = ALL<v...>::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.6 ALL<> Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

#### 16.6.1 Field Documentation

##### 16.6.1.1 value

```
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.7 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.

```
#include <field-traits.h>
```

### 16.7.1 Detailed Description

Arbitrary precision integers: GMP.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.8 AreEqual< X, Y > Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = false

### 16.8.1 Field Documentation

#### 16.8.1.1 value

```
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.9 AreEqual< X, X > Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = true

### 16.9.1 Field Documentation

#### 16.9.1.1 value

```
const bool value = true [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.10 Argument Struct Reference

```
#include <args-parser.h>
```

### Data Fields

- char [c](#)
- const char \* [example](#)
- const char \* [helpString](#)
- [ArgumentType](#) type
- void \* [data](#)

### 16.10.1 Field Documentation

#### 16.10.1.1 c

`char c`

#### 16.10.1.2 example

`const char* example`

#### 16.10.1.3 helpString

`const char* helpString`

#### 16.10.1.4 type

`ArgumentType type`

#### 16.10.1.5 data

`void* data`

The documentation for this struct was generated from the following file:

- [args-parser.h](#)

## 16.11 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [Field](#) [field](#)
- typedef [Field](#) & [type](#)

### 16.11.1 Member Typedef Documentation

#### 16.11.1.1 field

typedef [Field](#) [field](#)

#### 16.11.1.2 type

typedef [Field](#)& [type](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.12 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FFPACK::RNSInteger< RNS >](#) [field](#)
- typedef [FFPACK::RNSInteger< RNS >](#) [type](#)

### 16.12.1 Member Typedef Documentation

#### 16.12.1.1 field

```
typedef FFPACK::RNSInteger<RNS> field
```

#### 16.12.1.2 type

```
typedef FFPACK::RNSInteger<RNS> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.13 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

### 16.13.1 Member Typedef Documentation

#### 16.13.1.1 field

```
typedef Givaro::ZRing<T> field
```

#### 16.13.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.14 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

## 16.14.1 Member Typedef Documentation

### 16.14.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.14.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.15 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

## 16.15.1 Member Typedef Documentation

### 16.15.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.15.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.16 Auto Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.17 Bench< Elt > Class Template Reference

### Public Types

- using [Field](#) = Modular< Elt >
- using [Elt\\_ptr](#) = typename [Field::Element\\_ptr](#)
- using [Residu](#) = typename [Field::Residu\\_t](#)
- template<bool B, class T = void>  
using [enable\\_if\\_t](#) = typename std::enable\_if< B, T >::type

- `template<typename Simd >`  
`using is_same_element = typename Simd::template is_same_element< Field >`
- `template<typename E >`  
`using enable_if_no_simd_t = enable_if_t< Simd< E >::vect_size==1 >`
- `template<typename E >`  
`using enable_if_simd128_t = enable_if_t< sizeof(E) *Simd< E >::vect_size==16 >`
- `template<typename E >`  
`using enable_if_simd256_t = enable_if_t< sizeof(E) *Simd< E >::vect_size==32 >`
- `template<typename E >`  
`using enable_if_simd512_t = enable_if_t< sizeof(E) *Simd< E >::vect_size==64 >`

## Public Member Functions

- `Bench (size_t m, size_t n, size_t iters, bool inplace)`
- `template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * = nullptr>`  
`void doBenchs ()`
- `template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t< _E > * = nullptr>`  
`void run (bool allsimd)`

## Static Public Member Functions

- `template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>`  
`static Residu cardinality ()`
- `template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>`  
`static Residu cardinality ()`

## Protected Attributes

- `Field F`
- `const size_t m`
- `const size_t n`
- `const size_t iters`
- `const bool inplace`

## 16.17.1 Member Typedef Documentation

### 16.17.1.1 Field

`using Field = Modular<Elt>`

### 16.17.1.2 Elt\_ptr

`using Elt_ptr = typename Field::Element_ptr`

### 16.17.1.3 Residu

`using Residu = typename Field::Residu_t`

### 16.17.1.4 enable\_if\_t

`using enable_if_t = typename std::enable_if<B, T>::type`

**16.17.1.5 is\_same\_element**

```
using is_same_element = typename Simd::template is_same_element<Field>
```

**16.17.1.6 enable\_if\_no\_simd\_t**

```
using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
```

**16.17.1.7 enable\_if\_simd128\_t**

```
using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
```

**16.17.1.8 enable\_if\_simd256\_t**

```
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

**16.17.1.9 enable\_if\_simd512\_t**

```
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

**16.17.2 Constructor & Destructor Documentation****16.17.2.1 Bench()**

```
Bench (
    size_t m,
    size_t n,
    size_t iters,
    bool inplace ) [inline]
```

**16.17.3 Member Function Documentation****16.17.3.1 cardinality() [1/2]**

```
static Residu cardinality ( ) [inline], [static]
```

**16.17.3.2 cardinality() [2/2]**

```
static Residu cardinality ( ) [inline], [static]
```

**16.17.3.3 doBenchs()**

```
void doBenchs ( ) [inline]
```

**16.17.3.4 run()**

```
void run (
    bool allsimd ) [inline]
```

## 16.17.4 Field Documentation

### 16.17.4.1 F

`Field F` [protected]

### 16.17.4.2 m

`const size_t m` [protected]

### 16.17.4.3 n

`const size_t n` [protected]

### 16.17.4.4 iters

`const size_t iters` [protected]

### 16.17.4.5 inplace

`const bool inplace` [protected]

The documentation for this class was generated from the following file:

- [benchmark-storage-transpose.C](#)

## 16.18 Bini Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.19 Block Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.20 BlockTransposeSIMD< Field, Simd, > Struct Template Reference

```
#include <fflas_transpose.h>
```

### Public Member Functions

- `template<size_t s = Simd::vect_size, lsSimdSize<_s, 1> * = nullptr>`  
void [transpose](#) (const [Field](#) &F, ConstElement\_ptr A, size\_t lda, Element\_ptr B, size\_t ldb) const
- `template<size_t s = Simd::vect_size, lsSimdSize<_s, 2> * = nullptr>`  
void [transpose](#) (const [Field](#) &F, ConstElement\_ptr A, size\_t lda, Element\_ptr B, size\_t ldb) const
- `template<size_t s = Simd::vect_size, lsSimdSize<_s, 4> * = nullptr>`  
void [transpose](#) (const [Field](#) &F, ConstElement\_ptr A, size\_t lda, Element\_ptr B, size\_t ldb) const
- `template<size_t s = Simd::vect_size, lsSimdSize<_s, 8> * = nullptr>`  
void [transpose](#) (const [Field](#) &F, ConstElement\_ptr A, size\_t lda, Element\_ptr B, size\_t ldb) const
- `template<size_t s = Simd::vect_size, lsSimdSize<_s, 16> * = nullptr>`  
void [transpose](#) (const [Field](#) &F, ConstElement\_ptr A, size\_t lda, Element\_ptr B, size\_t ldb) const

## Static Public Member Functions

- static constexpr size\_t [size](#) ()
- static const std::string [info](#) ()

### 16.20.1 Member Function Documentation

#### 16.20.1.1 [size\(\)](#)

```
static constexpr size_t size ( ) [inline], [static], [constexpr]
```

#### 16.20.1.2 [info\(\)](#)

```
static const std::string info ( ) [inline], [static]
```

#### 16.20.1.3 [transpose\(\)](#) [1/5]

```
void transpose (
    const Field & F,
    ConstElement_ptr A,
    size_t lda,
    Element_ptr B,
    size_t ldb ) const [inline]
```

#### 16.20.1.4 [transpose\(\)](#) [2/5]

```
void transpose (
    const Field & F,
    ConstElement_ptr A,
    size_t lda,
    Element_ptr B,
    size_t ldb ) const [inline]
```

#### 16.20.1.5 [transpose\(\)](#) [3/5]

```
void transpose (
    const Field & F,
    ConstElement_ptr A,
    size_t lda,
    Element_ptr B,
    size_t ldb ) const [inline]
```

#### 16.20.1.6 [transpose\(\)](#) [4/5]

```
void transpose (
    const Field & F,
    ConstElement_ptr A,
    size_t lda,
    Element_ptr B,
    size_t ldb ) const [inline]
```

**16.20.1.7 transpose()** [5/5]

```
void transpose (
    const Field & F,
    ConstElement_ptr A,
    size_t lda,
    Element_ptr B,
    size_t ldb ) const [inline]
```

The documentation for this struct was generated from the following file:

- [fflas\\_transpose.h](#)

**16.21 callLUdivine\_small< Element > Class Template Reference****Public Member Functions**

- `template<class Field >`  
`size_t operator()` (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)

**16.21.1 Member Function Documentation****16.21.1.1 operator>()()**

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

**16.22 callLUdivine\_small< double > Class Reference****Public Member Functions**

- `template<class Field >`  
`size_t operator()` (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)

**16.22.1 Member Function Documentation****16.22.1.1 operator>()()**

```
size_t operator() (
    const Field & F,
```

```

const FFLAS::FFLAS_DIAG Diag,
const FFLAS::FFLAS_TRANSPOSE trans,
const size_t M,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
size_t * P,
size_t * Q,
const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.23 callLUdivine\_small< float > Class Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

### 16.23.1 Member Function Documentation

#### 16.23.1.1 operator()()

```

size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.24 CharpolyFailed Class Reference

```
#include <ffpack.h>
```

The documentation for this class was generated from the following file:

- [ffpack.h](#)

## 16.25 Checker\_Empty< Field > Struct Template Reference

```
#include <checker_empty.h>
```

### Public Member Functions

- `template<typename... Params>`  
`Checker_Empty (Params... parameters)`

- `template<typename... Params>`  
`bool check (Params... parameters)`

## 16.25.1 Constructor & Destructor Documentation

### 16.25.1.1 Checker\_Empty()

```
Checker_Empty (
    Params... parameters ) [inline]
```

## 16.25.2 Member Function Documentation

### 16.25.2.1 check()

```
bool check (
    Params... parameters ) [inline]
```

The documentation for this struct was generated from the following file:

- [checker\\_empty.h](#)

## 16.26 CheckerImplem\_charpoly< Field, Polynomial > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_charpoly](#) (const [Field](#) &F\_, const `size_t` n\_, typename [Field::ConstElement\\_ptr](#) A, `size_t` lda\_)
- [CheckerImplem\\_charpoly](#) (typename [Field::RandIter](#) &G, const `size_t` n\_, typename [Field::ConstElement\\_ptr](#) A, `size_t` lda\_)
- [~CheckerImplem\\_charpoly](#) ()
- `bool check` ([Polynomial](#) &g)

## 16.26.1 Constructor & Destructor Documentation

### 16.26.1.1 CheckerImplem\_charpoly() [1/2]

```
CheckerImplem_charpoly (
    const Field & F_,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

### 16.26.1.2 CheckerImplem\_charpoly() [2/2]

```
CheckerImplem_charpoly (
    typename Field::RandIter & G,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

### 16.26.1.3 ~CheckerImplem\_charpoly()

```
~CheckerImplem_charpoly ( ) [inline]
```

## 16.26.2 Member Function Documentation

### 16.26.2.1 check()

```
bool check (
    Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_charpoly.inl](#)

## 16.27 CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > Class Template Reference

### Public Types

- typedef Givaro::ZRing< Givaro::Integer > [Ring](#)

### Public Member Functions

- [CheckerImplem\\_charpoly](#) (const [Ring](#) &F\_, const size\_t n\_, typename [Ring::ConstElement\\_ptr](#) A, size\_t lda\_)
- [CheckerImplem\\_charpoly](#) (typename [Ring::RandIter](#) &G, const size\_t n\_, typename [Ring::ConstElement\\_ptr](#) A, size\_t lda\_)
- [~CheckerImplem\\_charpoly](#) ()
- bool [check](#) (Polynomial &g)

### 16.27.1 Member Typedef Documentation

#### 16.27.1.1 Ring

```
typedef Givaro::ZRing<Givaro::Integer> Ring
```

### 16.27.2 Constructor & Destructor Documentation

#### 16.27.2.1 CheckerImplem\_charpoly() [1/2]

```
CheckerImplem_charpoly (
    const Ring & F_,
    const size_t n_,
    typename Ring::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

**16.27.2.2 CheckerImplem\_charpoly() [2/2]**

```
CheckerImplem_charpoly (
    typename Ring::RandIter & G,
    const size_t n_,
    typename Ring::ConstElement_ptr A,
    size_t lda_ ) [inline]
```

**16.27.2.3 ~CheckerImplem\_charpoly()**

```
~CheckerImplem_charpoly ( ) [inline]
```

**16.27.3 Member Function Documentation****16.27.3.1 check()**

```
bool check (
    Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_charpoly.inl](#)

**16.28 CheckerImplem\_Det< Field > Class Template Reference****Public Member Functions**

- [CheckerImplem\\_Det](#) (const [Field](#) &F\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_Det](#) (typename [Field::RandIter](#) &G, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_Det](#) ()
- bool [check](#) (const typename [Field::Element](#) &det, typename [Field::ConstElement\\_ptr](#) LU, size\_t lda, size\_t \*P, size\_t \*Q) const

*check if the Det factorization is correct.*

**16.28.1 Constructor & Destructor Documentation****16.28.1.1 CheckerImplem\_Det() [1/2]**

```
CheckerImplem_Det (
    const Field & F_,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

**16.28.1.2 CheckerImplem\_Det() [2/2]**

```
CheckerImplem_Det (
    typename Field::RandIter & G,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

### 16.28.1.3 ~CheckerImplem\_Det()

```
~CheckerImplem_Det ( ) [inline]
```

## 16.28.2 Member Function Documentation

### 16.28.2.1 check()

```
bool check (
    const typename Field::Element & det,
    typename Field::ConstElement_ptr LU,
    size_t lda,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the Det factorization is correct.

Needs matrix in LU form

#### Parameters

<i>LU,storage</i>	for L and U
<i>det</i>	
<i>P</i>	
<i>Q</i>	

The documentation for this class was generated from the following file:

- [checker\\_det.inl](#)

## 16.29 CheckerImplem\_fgemm< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_fgemm](#) (const [Field](#) &F\_, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [CheckerImplem\\_fgemm](#) (typename [Field::RandIter](#) &G, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [~CheckerImplem\\_fgemm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C)

### 16.29.1 Constructor & Destructor Documentation

#### 16.29.1.1 CheckerImplem\_fgemm() [1/2]

```
CheckerImplem_fgemm (
    const Field & F_,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc_ ) [inline]
```

## 16.29.1.2 CheckerImplem\_fgemm() [2/2]

```
CheckerImplem_fgemm (
    typename Field::RandIter & G,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc_ ) [inline]
```

## 16.29.1.3 ~CheckerImplem\_fgemm()

```
~CheckerImplem_fgemm ( ) [inline]
```

## 16.29.2 Member Function Documentation

## 16.29.2.1 check()

```
bool check (
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_fgemm.inl](#)

## 16.30 CheckerImplem\_ftrsm&lt; Field &gt; Class Template Reference

## Public Member Functions

- [CheckerImplem\\_ftrsm](#) (const [Field](#) &F\_, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [CheckerImplem\\_ftrsm](#) (typename [Field::RandIter](#) &G, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [~CheckerImplem\\_ftrsm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) uplo, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, size\_t ldx)

## 16.30.1 Constructor &amp; Destructor Documentation

## 16.30.1.1 CheckerImplem\_ftrsm() [1/2]

```
CheckerImplem_ftrsm (
    const Field & F_,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
```

```
const typename Field::ConstElement_ptr B,
const size_t ldb ) [inline]
```

### 16.30.1.2 CheckerImplem\_ftsrsm() [2/2]

```
CheckerImplem_ftsrsm (
    typename Field::RandIter & G,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

### 16.30.1.3 ~CheckerImplem\_ftsrsm()

```
~CheckerImplem_ftsrsm ( ) [inline]
```

## 16.30.2 Member Function Documentation

### 16.30.2.1 check()

```
bool check (
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO uplo,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const FFLAS::FFLAS_DIAG diag,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    const typename Field::ConstElement_ptr X,
    size_t ldx ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_ftsrsm.inl](#)

## 16.31 CheckerImplem\_invert< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_invert](#) (const [Field](#) &F\_, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [CheckerImplem\\_invert](#) (typename [Field::RandIter](#) &G, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [~CheckerImplem\\_invert](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, int nullity)

### 16.31.1 Constructor & Destructor Documentation

**16.31.1.1 CheckerImplem\_invert() [1/2]**

```
CheckerImplem_invert (
    const Field & F_,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

**16.31.1.2 CheckerImplem\_invert() [2/2]**

```
CheckerImplem_invert (
    typename Field::RandIter & G,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

**16.31.1.3 ~CheckerImplem\_invert()**

```
~CheckerImplem_invert ( ) [inline]
```

**16.31.2 Member Function Documentation****16.31.2.1 check()**

```
bool check (
    typename Field::ConstElement_ptr A,
    int nullity ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_invert.inl](#)

**16.32 CheckerImplem\_PLUQ< Field > Class Template Reference****Public Member Functions**

- [CheckerImplem\\_PLUQ](#) (const [Field](#) &F\_, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_PLUQ](#) (typename [Field::RandIter](#) &G, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_PLUQ](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, size\_t lda, const [FFLAS::FFLAS\\_DIAG](#) Diag, size\_t r, size\_t \*P, size\_t \*Q) const  
*check if the PLUQ factorization is correct.*

**16.32.1 Constructor & Destructor Documentation****16.32.1.1 CheckerImplem\_PLUQ() [1/2]**

```
CheckerImplem_PLUQ (
    const Field & F_,
    size_t m_,
    size_t n_,
```

```

typename Field::ConstElement_ptr A,
size_t lda ) [inline]

```

### 16.32.1.2 CheckerImplem\_PLUQ() [2/2]

```

CheckerImplem_PLUQ (
    typename Field::RandIter & G,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]

```

### 16.32.1.3 ~CheckerImplem\_PLUQ()

```

~CheckerImplem_PLUQ ( ) [inline]

```

## 16.32.2 Member Function Documentation

### 16.32.2.1 check()

```

bool check (
    typename Field::ConstElement_ptr A,
    size_t lda,
    const FFLAS::FFLAS_DIAG Diag,
    size_t r,
    size_t * P,
    size_t * Q ) const [inline]

```

check if the PLUQ factorization is correct.

Returns true if  $w - P(L(U(Q.v))) == 0$

#### Parameters

$A$	
$r$	
$P$	
$Q$	

The documentation for this class was generated from the following file:

- [checker\\_pluq.inl](#)

## 16.33 Classic Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.34 Column Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.35 CompactElement< Element > Struct Template Reference

### Public Types

- typedef Element [type](#)

### 16.35.1 Member Typedef Documentation

#### 16.35.1.1 type

typedef Element [type](#)

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.36 CompactElement< double > Struct Reference

### Public Types

- typedef int32\_t [type](#)

### 16.36.1 Member Typedef Documentation

#### 16.36.1.1 type

typedef int32\_t [type](#)

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.37 CompactElement< float > Struct Reference

### Public Types

- typedef int16\_t [type](#)

### 16.37.1 Member Typedef Documentation

#### 16.37.1.1 type

typedef int16\_t [type](#)

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.38 CompactElement< int16\_t > Struct Reference

### Public Types

- typedef int8\_t [type](#)

### 16.38.1 Member Typedef Documentation

#### 16.38.1.1 type

```
typedef int8_t type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.39 CompactElement< int32\_t > Struct Reference

### Public Types

- typedef int16\_t [type](#)

### 16.39.1 Member Typedef Documentation

#### 16.39.1.1 type

```
typedef int16_t type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.40 CompactElement< int64\_t > Struct Reference

### Public Types

- typedef int32\_t [type](#)

### 16.40.1 Member Typedef Documentation

#### 16.40.1.1 type

```
typedef int32_t type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.41 compatible\_data\_type< Field > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.41.1 Field Documentation

#### 16.41.1.1 value

```
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.42 compatible\_data\_type< Givaro::ZRing< double > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.42.1 Field Documentation

#### 16.42.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.43 compatible\_data\_type< Givaro::ZRing< float > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.43.1 Field Documentation

#### 16.43.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.44 Compose< H1, H2 > Struct Template Reference

### Public Member Functions

- [Compose](#) ()
- [Compose](#) (const [Compose](#) &other)
- [Compose](#) (const [Sequential](#) &S)
- [Compose](#) (size\_t th1, size\_t th2)
- [Compose](#) (const H1 &o1, const H2 &o2)
- H1 [first\\_component](#) () const
- H2 [second\\_component](#) () const

### Friends

- std::ostream & [operator<<](#) (std::ostream &o, const [Compose](#) &c)

### 16.44.1 Constructor & Destructor Documentation

**16.44.1.1 Compose()** [1/5]

```
Compose ( ) [inline]
```

**16.44.1.2 Compose()** [2/5]

```
Compose (
    const Compose< H1, H2 > & other ) [inline]
```

**16.44.1.3 Compose()** [3/5]

```
Compose (
    const Sequential & S ) [inline]
```

**16.44.1.4 Compose()** [4/5]

```
Compose (
    size_t th1,
    size_t th2 ) [inline]
```

**16.44.1.5 Compose()** [5/5]

```
Compose (
    const H1 & o1,
    const H2 & o2 ) [inline]
```

**16.44.2 Member Function Documentation****16.44.2.1 first\_component()**

```
H1 first_component ( ) const [inline]
```

**16.44.2.2 second\_component()**

```
H2 second_component ( ) const [inline]
```

**16.44.3 Friends And Related Function Documentation****16.44.3.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & o,
    const Compose< H1, H2 > & c ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.45 Simd128\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.45.1 Field Documentation

##### 16.45.1.1 v

[vect\\_t v](#)

##### 16.45.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.46 Simd128\_impl< true, true, false, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.46.1 Field Documentation

##### 16.46.1.1 v

[vect\\_t v](#)

##### 16.46.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.47 Simd128\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.47.1 Field Documentation

**16.47.1.1 v**

`vect_t v`

**16.47.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)

**16.48 Simd128\_impl< true, true, true, 2 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.48.1 Field Documentation****16.48.1.1 v**

`vect_t v`

**16.48.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

**16.49 Simd128\_impl< true, true, true, 4 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.49.1 Field Documentation****16.49.1.1 v**

`vect_t v`

**16.49.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.50 Simd128\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.50.1 Field Documentation

##### 16.50.1.1 v

[vect\\_t v](#)

##### 16.50.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)

## 16.51 Simd256\_impl< true, false, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.51.1 Field Documentation

##### 16.51.1.1 v

[vect\\_t v](#)

##### 16.51.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd256\\_double.inl](#)

## 16.52 Simd256\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.52.1 Field Documentation

**16.52.1.1 v**`vect_t v`**16.52.1.2 t**`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

**16.53 Simd256\_impl< true, true, false, 4 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.53.1 Field Documentation****16.53.1.1 v**`vect_t v`**16.53.1.2 t**`scalar_t t`

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

**16.54 Simd256\_impl< true, true, false, 8 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.54.1 Field Documentation****16.54.1.1 v**`vect_t v`**16.54.1.2 t**`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

## 16.55 Simd256\_impl< true, true, true, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.55.1 Field Documentation

##### 16.55.1.1 v

[vect\\_t v](#)

##### 16.55.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.56 Simd256\_impl< true, true, true, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.56.1 Field Documentation

##### 16.56.1.1 v

[vect\\_t v](#)

##### 16.56.1.2 t

[scalar\\_t t](#)

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.57 Simd256\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.57.1 Field Documentation

**16.57.1.1 v**

`vect_t v`

**16.57.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

**16.58 Simd512\_impl< true, true, false, 8 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.58.1 Field Documentation****16.58.1.1 v**

`vect_t v`

**16.58.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

**16.59 Simd512\_impl< true, true, true, 8 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.59.1 Field Documentation****16.59.1.1 v**

`vect_t v`

**16.59.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

## 16.60 ConvertTo< T > Struct Template Reference

Force conversion to appropriate element type of ElementCategory T.

```
#include <field-traits.h>
```

### 16.60.1 Detailed Description

```
template<class T>
```

```
struct FFLAS::ModeCategories::ConvertTo< T >
```

Force conversion to appropriate element type of ElementCategory T.

e.g.

- ConvertTo<ElementCategories::MachineFloatTag> tries conversion of Modular<int> to Modular<double>
- ConvertTo<ElementCategories::FixedPreIntTag> tries conversion of Modular<Integer> to Modular<RecInt<K>>
- ConvertTo<ElementCategories::ArbitraryPreIntTag> tries conversion of Modular<Integer> to RNSInteger

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.61 Coo< ValT, IdxT > Struct Template Reference

### Public Types

- using [Self](#) = [Coo](#)< ValT, IdxT >

### Public Member Functions

- [Coo](#) (ValT v, IdxT r, IdxT c)
- [Coo](#) ()=default
- [Coo](#) (const [Self](#) &)=default
- [Coo](#) ([Self](#) &&)=default
- [Self](#) & [operator](#)= (const [Self](#) &)=default
- [Self](#) & [operator](#)= ([Self](#) &&)=default

### Data Fields

- ValT [val](#) = 0
- IdxT [row](#) = 0
- IdxT [col](#) = 0

### 16.61.1 Member Typedef Documentation

#### 16.61.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

### 16.61.2 Constructor & Destructor Documentation

**16.61.2.1** **Coo()** [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

**16.61.2.2** **Coo()** [2/4]

```
Coo ( ) [default]
```

**16.61.2.3** **Coo()** [3/4]

```
Coo (
    const Self & ) [default]
```

**16.61.2.4** **Coo()** [4/4]

```
Coo (
    Self && ) [default]
```

**16.61.3** **Member Function Documentation****16.61.3.1** **operator=()** [1/2]

```
Self& operator= (
    const Self & ) [default]
```

**16.61.3.2** **operator=()** [2/2]

```
Self& operator= (
    Self && ) [default]
```

**16.61.4** **Field Documentation****16.61.4.1** **val**

```
ValT val = 0
```

**16.61.4.2** **row**

```
IdxT row = 0
```

**16.61.4.3** **col**

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

## 16.62 `Coo< Field > Struct` Template Reference

```
#include <read_sparse.h>
```

### Public Member Functions

- `Coo` ()=default
- `Coo` (typename `Field::Element` v, `index_t` r, `index_t` c)
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

### Data Fields

- `Field::Element` val = 0
- `index_t` col = 0
- `index_t` row = 0
- bool `deleted` = false

### 16.62.1 Constructor & Destructor Documentation

#### 16.62.1.1 `Coo`() [1/4]

```
Coo ( ) [default]
```

#### 16.62.1.2 `Coo`() [2/4]

```
Coo (
    typename Field::Element v,
    index_t r,
    index_t c ) [inline]
```

#### 16.62.1.3 `Coo`() [3/4]

```
Coo (
    const Self & ) [default]
```

#### 16.62.1.4 `Coo`() [4/4]

```
Coo (
    Self && ) [default]
```

### 16.62.2 Member Function Documentation

#### 16.62.2.1 `operator=`() [1/2]

```
Self& operator= (
    const Self & ) [default]
```

**16.62.2.2 operator=()** [2/2]

```
Self& operator= (
    Self && ) [default]
```

**16.62.3 Field Documentation****16.62.3.1 val**

```
Field::Element val = 0
```

**16.62.3.2 col**

```
index_t col = 0
```

**16.62.3.3 row**

```
index_t row = 0
```

**16.62.3.4 deleted**

```
bool deleted = false
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

**16.63  $\text{Coo} < \text{ValT}, \text{IdxT} >$  Struct Template Reference****Public Types**

- using [Self](#) = [Coo](#) < ValT, IdxT >

**Public Member Functions**

- [Coo](#) (ValT v, IdxT r, IdxT c)
- [Coo](#) ()=default
- [Coo](#) (const [Self](#) &)=default
- [Coo](#) ([Self](#) &&)=default
- [Self](#) & [operator](#)= (const [Self](#) &)=default
- [Self](#) & [operator](#)= ([Self](#) &&)=default

**Data Fields**

- ValT [val](#) = 0
- IdxT [row](#) = 0
- IdxT [col](#) = 0

**16.63.1 Member Typedef Documentation****16.63.1.1 Self**

```
using Self = Coo<ValT, IdxT>
```

## 16.63.2 Constructor & Destructor Documentation

### 16.63.2.1 `Coo()` [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

### 16.63.2.2 `Coo()` [2/4]

```
Coo ( ) [default]
```

### 16.63.2.3 `Coo()` [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.63.2.4 `Coo()` [4/4]

```
Coo (
    Self && ) [default]
```

## 16.63.3 Member Function Documentation

### 16.63.3.1 `operator=()` [1/2]

```
Self& operator= (
    const Self & ) [default]
```

### 16.63.3.2 `operator=()` [2/2]

```
Self& operator= (
    Self && ) [default]
```

## 16.63.4 Field Documentation

### 16.63.4.1 `val`

```
ValT val = 0
```

### 16.63.4.2 `row`

```
IdxT row = 0
```

### 16.63.4.3 col

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

## 16.64 CooMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

### Data Fields

- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int16\\_t > \\* \\_coo16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int32\\_t > \\* \\_coo32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int64\\_t > \\* \\_coo64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int16\\_t > \\* \\_coo16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int32\\_t > \\* \\_coo32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int64\\_t > \\* \\_coo64\\_zo](#) = nullptr

### 16.64.1 Field Documentation

#### 16.64.1.1 \_coo16

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int16_t>* _coo16 = nullptr
```

#### 16.64.1.2 \_coo32

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int32_t>* _coo32 = nullptr
```

#### 16.64.1.3 \_coo64

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int64_t>* _coo64 = nullptr
```

#### 16.64.1.4 \_coo16\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int16_t>* _coo16_zo = nullptr
```

#### 16.64.1.5 \_coo32\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int32_t>* _coo32_zo = nullptr
```

#### 16.64.1.6 \_coo64\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int64_t>* _coo64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.65 `count_nonconst_lvalue_reference< T >` Struct Template Reference

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.66 `count_nonconst_lvalue_reference< const T &, O... >` Struct Template Reference

### Static Public Attributes

- static constexpr size\_t `n` = `count_nonconst_lvalue_reference<O...>::n`

### 16.66.1 Field Documentation

#### 16.66.1.1 `n`

```
constexpr size_t n = count_nonconst_lvalue_reference<O...>::n [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.67 `count_nonconst_lvalue_reference< T &, O... >` Struct Template Reference

### Static Public Attributes

- static constexpr size\_t `n`

### 16.67.1 Field Documentation

#### 16.67.1.1 `n`

```
constexpr size_t n [static], [constexpr]
```

**Initial value:**

```
= std::integral_constant<size_t, 1>::value
+ count_nonconst_lvalue_reference<O...>::n
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.68 `count_nonconst_lvalue_reference< T, O... >` Struct Template Reference

### Static Public Attributes

- static constexpr size\_t `n` = `count_nonconst_lvalue_reference<O...>::n`

### 16.68.1 Field Documentation

### 16.68.1.1 `n`

```
constexpr size_t n = count_nonconst_lvalue_reference<0...>::n [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.69 `count_nonconst_lvalue_reference<>` Struct Reference

### Static Public Attributes

- static constexpr size\_t `n` = std::integral\_constant<size\_t, 0>::value

### 16.69.1 Field Documentation

#### 16.69.1.1 `n`

```
constexpr size_t n = std::integral_constant<size_t, 0>::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.70 `CsrMat<Field>` Struct Template Reference

```
#include <fflas_sparse.h>
```

### Data Fields

- `FFLAS::Sparse<Field, SparseMatrix_t::CSR, int16_t> * _csr16` = nullptr
- `FFLAS::Sparse<Field, SparseMatrix_t::CSR, int32_t> * _csr32` = nullptr
- `FFLAS::Sparse<Field, SparseMatrix_t::CSR, int64_t> * _csr64` = nullptr
- `FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int16_t> * _csr16_zo` = nullptr
- `FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int32_t> * _csr32_zo` = nullptr
- `FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int64_t> * _csr64_zo` = nullptr

### 16.70.1 Field Documentation

#### 16.70.1.1 `_csr16`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int16_t>* _csr16 = nullptr
```

#### 16.70.1.2 `_csr32`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int32_t>* _csr32 = nullptr
```

#### 16.70.1.3 `_csr64`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int64_t>* _csr64 = nullptr
```

#### 16.70.1.4 `_csr16_zo`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int16_t>* _csr16_zo = nullptr
```

#### 16.70.1.5 `_csr32_zo`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int32_t>* _csr32_zo = nullptr
```

#### 16.70.1.6 `_csr64_zo`

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int64_t>* _csr64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.71 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.

```
#include <field-traits.h>
```

### 16.71.1 Detailed Description

Use standard field operations, but keeps track of bounds on input and output.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.72 DefaultTag Struct Reference

No specific mode of action: use standard field operations.

```
#include <field-traits.h>
```

### 16.72.1 Detailed Description

No specific mode of action: use standard field operations.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.73 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.

```
#include <field-traits.h>
```

### 16.73.1 Detailed Description

Performs field operations with delayed mod reductions. Ensures result is reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.74 DivideAndConquer Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.75 ElementTraits< Element > Struct Template Reference

[ElementTraits](#).

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::GenericTag](#) value

#### 16.75.1 Detailed Description

```
template<class Element>
```

```
struct FFLAS::ElementTraits< Element >
```

[ElementTraits](#).

#### 16.75.2 Member Typedef Documentation

##### 16.75.2.1 value

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.76 ElementTraits< double > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

#### 16.76.1 Member Typedef Documentation

##### 16.76.1.1 value

```
typedef ElementCategories::MachineFloatTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.77 ElementTraits< FFPACK::rns\_double\_elt > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::RNSElementTag](#) value

#### 16.77.1 Member Typedef Documentation

### 16.77.1.1 value

typedef [ElementCategories::RNSElementTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.78 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

### 16.78.1 Member Typedef Documentation

#### 16.78.1.1 value

typedef [ElementCategories::MachineFloatTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.79 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::ArbitraryPrecIntTag](#) value

### 16.79.1 Member Typedef Documentation

#### 16.79.1.1 value

typedef [ElementCategories::ArbitraryPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.80 ElementTraits< int16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.80.1 Member Typedef Documentation

### 16.80.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.81 ElementTraits< int32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.81.1 Member Typedef Documentation

#### 16.81.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.82 ElementTraits< int64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.82.1 Member Typedef Documentation

#### 16.82.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.83 ElementTraits< int8\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.83.1 Member Typedef Documentation

### 16.83.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.84 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.84.1 Member Typedef Documentation

#### 16.84.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.85 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.85.1 Member Typedef Documentation

#### 16.85.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.86 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.86.1 Member Typedef Documentation

### 16.86.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.87 ElementTraits< uint16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.87.1 Member Typedef Documentation

#### 16.87.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.88 ElementTraits< uint32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.88.1 Member Typedef Documentation

#### 16.88.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.89 ElementTraits< uint64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.89.1 Member Typedef Documentation

**16.89.1.1 value**

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.90 ElementTraits< uint8\_t > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [ElementCategories::MachineIntTag](#) value

**16.90.1 Member Typedef Documentation****16.90.1.1 value**

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.91 EIMat< Field > Struct Template Reference**

```
#include <fflas_sparse.h>
```

**Data Fields**

- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL, int16\\_t > \\* \\_ell16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL, int32\\_t > \\* \\_ell32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL, int64\\_t > \\* \\_ell64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL\\_ZO, int16\\_t > \\* \\_ell16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL\\_ZO, int32\\_t > \\* \\_ell32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::ELL\\_ZO, int64\\_t > \\* \\_ell64\\_zo](#) = nullptr

**16.91.1 Field Documentation****16.91.1.1 \_ell16**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int16_t>* _ell16 = nullptr
```

**16.91.1.2 \_ell32**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int32_t>* _ell32 = nullptr
```

**16.91.1.3 \_ell64**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int64_t>* _ell64 = nullptr
```

**16.91.1.4    `_ell16_zo`**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int16_t>* _ell16_zo = nullptr
```

**16.91.1.5    `_ell32_zo`**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int32_t>* _ell32_zo = nullptr
```

**16.91.1.6    `_ell64_zo`**

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int64_t>* _ell64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

**16.92    Failure Class Reference**

A precondition failed.

```
#include <debug.h>
```

**Public Member Functions**

- [Failure](#) ()
- void [operator](#)() (const char \*function, int line, const char \*check)
- void [operator](#)() (const char \*function, const char \*file, int line, const char \*check)
- void [setErrorStream](#) (std::ostream &stream)
- std::ostream & [print](#) (std::ostream &o) const

**Protected Attributes**

- std::ostream \* [\\_errorStream](#)

**16.92.1    Detailed Description**

A precondition failed.

The `throw` mechanism is usually used here as in

```
if (!check)
    failure()(__func__, __LINE__, "this check just failed");
```

The parameters of the constructor help debugging.

**16.92.2    Constructor & Destructor Documentation****16.92.2.1    Failure()**

```
Failure ( ) [inline]
```

**16.92.3    Member Function Documentation**

**16.92.3.1 operator>() [1/2]**

```
void operator() (
    const char * function,
    int line,
    const char * check ) [inline]
```

A precondition failed.

**Parameters**

<i>function</i>	usually <b>func</b> , the function that threw the error
<i>line</i>	usually <b>LINE</b> , the line where it happened
<i>check</i>	a string telling what failed.

**16.92.3.2 operator>() [2/2]**

```
void operator() (
    const char * function,
    const char * file,
    int line,
    const char * check ) [inline]
```

A precondition failed. The parameter help debugging. This is not much different from the previous except we can digg faster in the file where the exception was triggered.

**Parameters**

<i>function</i>	usually <b>func</b> , the function that threw the error
<i>file</i>	usually <b>FILE</b> , the file where this function is
<i>line</i>	usually <b>LINE</b> , the line where it happened
<i>check</i>	a string telling what failed.

**16.92.3.3 setErrorMessage()**

```
void setErrorMessage (
    std::ostream & stream )
```

**16.92.3.4 print()**

```
std::ostream& print (
    std::ostream & o ) const [inline]
```

overload the virtual print of LinboxError.

**Parameters**

<i>o</i>	output stream
----------	---------------

**16.92.4 Field Documentation**

#### 16.92.4.1 `_errorStream`

```
std::ostream* _errorStream [protected]
```

The documentation for this class was generated from the following file:

- [debug.h](#)

### 16.93 FailureCharpolyCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

### 16.94 FailureDetCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

### 16.95 FailureFgemmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

### 16.96 FailureInvertCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

### 16.97 FailurePLUQCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

### 16.98 FailureTrsmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

### 16.99 FieldSimd< `_Field` > Class Template Reference

#### Public Types

- using `Field` = `_Field`
- using `Element` = typename `Field::Element`
- using `simd` = `Simd`< typename `_Field::Element` >
- using `vect_t` = typename `simd::vect_t`
- using `scalar_t` = typename `simd::scalar_t`

## Public Member Functions

- [FieldSimd](#) (const [Field](#) &f)
- [FieldSimd](#) (const [Self](#) &)=default
- [FieldSimd](#) ([Self](#) &&)=default
- [Self](#) & [operator=](#) (const [Self](#) &)=default
- [Self](#) & [operator=](#) ([Self](#) &&)=default
- [INLINE vect\\_t init](#) ([vect\\_t](#) &x, const [vect\\_t](#) a) const
- [INLINE vect\\_t init](#) (const [vect\\_t](#) a) const
- [INLINE vect\\_t add](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t addin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t subin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t zero](#) ([vect\\_t](#) &x) const
- [INLINE vect\\_t zero](#) () const
- [INLINE vect\\_t mod](#) ([vect\\_t](#) &c) const
- [INLINE vect\\_t mul](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mulin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpy](#) ([vect\\_t](#) &r, const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) c) const
- [INLINE vect\\_t axpy](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpyin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpy\\_r](#) ([vect\\_t](#) &r, const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) c) const
- [INLINE vect\\_t axpy\\_r](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpyin\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t maxpy](#) ([vect\\_t](#) &r, const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) c) const
- [INLINE vect\\_t maxpy](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t maxpyin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const

## Static Public Attributes

- static constexpr const size\_t [vect\\_size](#) = simd::vect\_size
- static constexpr const size\_t [alignment](#) = simd::alignment

### 16.99.1 Member Typedef Documentation

#### 16.99.1.1 Field

```
using Field = _Field
```

#### 16.99.1.2 Element

```
using Element = typename Field::Element
```

**16.99.1.3 simd**

```
using simd = Simd<typename _Field::Element>
```

**16.99.1.4 vect\_t**

```
using vect_t = typename simd::vect_t
```

**16.99.1.5 scalar\_t**

```
using scalar_t = typename simd::scalar_t
```

**16.99.2 Constructor & Destructor Documentation****16.99.2.1 FieldSimd() [1/3]**

```
FieldSimd (
    const Field & f ) [inline]
```

**16.99.2.2 FieldSimd() [2/3]**

```
FieldSimd (
    const Self & ) [default]
```

**16.99.2.3 FieldSimd() [3/3]**

```
FieldSimd (
    Self && ) [default]
```

**16.99.3 Member Function Documentation****16.99.3.1 operator=() [1/2]**

```
Self& operator= (
    const Self & ) [default]
```

**16.99.3.2 operator=() [2/2]**

```
Self& operator= (
    Self && ) [default]
```

**16.99.3.3 init() [1/2]**

```
INLINE vect_t init (
    vect_t & x,
    const vect_t a ) const [inline]
```

**16.99.3.4 init()** [2/2]

```
INLINE vect_t init (
    const vect_t a ) const [inline]
```

**16.99.3.5 add()** [1/2]

```
INLINE vect_t add (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

**16.99.3.6 add()** [2/2]

```
INLINE vect_t add (
    const vect_t a,
    const vect_t b ) [inline]
```

**16.99.3.7 addin()**

```
INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) const [inline]
```

**16.99.3.8 add\_r()** [1/2]

```
INLINE vect_t add_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.9 add\_r()** [2/2]

```
INLINE vect_t add_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.10 addin\_r()**

```
INLINE vect_t addin_r (
    vect_t & a,
    const vect_t b ) const [inline]
```

**16.99.3.11 sub()** [1/2]

```
INLINE vect_t sub (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

**16.99.3.12 sub()** [2/2]

```
INLINE vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline]
```

**16.99.3.13 subin()**

```
INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.99.3.14 sub\_r()** [1/2]

```
INLINE vect_t sub_r (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.99.3.15 sub\_r()** [2/2]

```
INLINE vect_t sub_r (  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.99.3.16 subin\_r()**

```
INLINE vect_t subin_r (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.99.3.17 zero()** [1/2]

```
INLINE vect_t zero (  
    vect_t & x ) const [inline]
```

**16.99.3.18 zero()** [2/2]

```
INLINE vect_t zero ( ) const [inline]
```

**16.99.3.19 mod()**

```
INLINE vect_t mod (  
    vect_t & c ) const [inline]
```

**16.99.3.20 mul()** [1/2]

```
INLINE vect_t mul (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.99.3.21 mul()** [2/2]

```
INLINE vect_t mul (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.22 mulin()**

```
INLINE vect_t mulin (
    vect_t & a,
    const vect_t b ) const [inline]
```

**16.99.3.23 mul\_r()** [1/2]

```
INLINE vect_t mul_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.24 mul\_r()** [2/2]

```
INLINE vect_t mul_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.25 axpy()** [1/2]

```
INLINE vect_t axpy (
    vect_t & r,
    const vect_t a,
    const vect_t b,
    const vect_t c ) const [inline]
```

**16.99.3.26 axpy()** [2/2]

```
INLINE vect_t axpy (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.27 axpyin()**

```
INLINE vect_t axpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.99.3.28 axpy\_r()** [1/2]

```
INLINE vect_t axpy_r (
    vect_t & r,
    const vect_t a,
```

```
const vect_t b,
const vect_t c ) const [inline]
```

#### 16.99.3.29 axpy\_r() [2/2]

```
INLINE vect_t axpy_r (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

#### 16.99.3.30 axpyin\_r()

```
INLINE vect_t axpyin_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

#### 16.99.3.31 maxpy() [1/2]

```
INLINE vect_t maxpy (
    vect_t & r,
    const vect_t a,
    const vect_t b,
    const vect_t c ) const [inline]
```

#### 16.99.3.32 maxpy() [2/2]

```
INLINE vect_t maxpy (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

#### 16.99.3.33 maxpyin()

```
INLINE vect_t maxpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

### 16.99.4 Field Documentation

#### 16.99.4.1 vect\_size

```
constexpr const size_t vect_size = simd::vect_size [static], [constexpr]
```

#### 16.99.4.2 alignment

```
constexpr const size_t alignment = simd::alignment [static], [constexpr]
```

The documentation for this class was generated from the following file:

- [simd\\_modular.inl](#)

## 16.100 FieldTraits< Field > Struct Template Reference

FieldTrait.

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::GenericTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

#### 16.100.1 Detailed Description

```
template<class Field>
```

```
struct FFLAS::FieldTraits< Field >
```

FieldTrait.

#### 16.100.2 Member Typedef Documentation

##### 16.100.2.1 category

```
typedef FieldCategories::GenericTag category
```

#### 16.100.3 Field Documentation

##### 16.100.3.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.101 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

#### 16.101.1 Member Typedef Documentation

##### 16.101.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

## 16.101.2 Field Documentation

### 16.101.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.102 `FieldTraits< FFPACK::RNSIntegerMod< T > >` Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::ModularTag](#) `category`

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.102.1 Member Typedef Documentation

### 16.102.1.1 `category`

```
typedef FieldCategories::ModularTag category
```

## 16.102.2 Field Documentation

### 16.102.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.103 `FieldTraits< Givaro::Modular< Element > >` Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::ModularTag](#) `category`

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.103.1 Member Typedef Documentation

#### 16.103.1.1 category

typedef [FieldCategories::ModularTag](#) category

### 16.103.2 Field Documentation

#### 16.103.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.104 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::ModularTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = true

### 16.104.1 Member Typedef Documentation

#### 16.104.1.1 category

typedef [FieldCategories::ModularTag](#) category

### 16.104.2 Field Documentation

#### 16.104.2.1 balanced

```
const bool balanced = true [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.105 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.105.1 Member Typedef Documentation

#### 16.105.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.105.2 Field Documentation

#### 16.105.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.106 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.106.1 Member Typedef Documentation

#### 16.106.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.106.2 Field Documentation

#### 16.106.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.107 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.107.1 Member Typedef Documentation

#### 16.107.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.107.2 Field Documentation

#### 16.107.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.108 FieldTraits< Givaro::ZRing< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.108.1 Member Typedef Documentation

#### 16.108.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.108.2 Field Documentation

#### 16.108.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.109 FieldTraits< Givaro::ZRing< int32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.109.1 Member Typedef Documentation

#### 16.109.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.109.2 Field Documentation

#### 16.109.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.110 FieldTraits< Givaro::ZRing< int64\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.110.1 Member Typedef Documentation

#### 16.110.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.110.2 Field Documentation

### 16.110.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.111 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.111.1 Member Typedef Documentation

#### 16.111.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.111.2 Field Documentation

#### 16.111.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.112 FieldTraits< Givaro::ZRing< uint16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.112.1 Member Typedef Documentation

#### 16.112.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

## 16.112.2 Field Documentation

### 16.112.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.113 `FieldTraits< Givaro::ZRing< uint32_t > >` Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) `category`

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.113.1 Member Typedef Documentation

### 16.113.1.1 `category`

```
typedef FieldCategories::UnparametricTag category
```

## 16.113.2 Field Documentation

### 16.113.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.114 `FieldTraits< Givaro::ZRing< uint64_t > >` Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) `category`

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.114.1 Member Typedef Documentation

**16.114.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

**16.114.2 Field Documentation****16.114.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.115 Fixed Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.116 FixedPrecIntTag Struct Reference**

Fixed precision integers above machine precision: Givaro::reclnt.

```
#include <field-traits.h>
```

**16.116.1 Detailed Description**

Fixed precision integers above machine precision: Givaro::reclnt.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.117 ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >::FloatingPointTestDistribution Class Reference

**Public Types**

- using [IntType](#) = typename make\_unsigned\_int< Element >::type

**Public Member Functions**

- [FloatingPointTestDistribution](#) ()
- template<class Generator >  
[Element operator\(\)](#) (Generator &g)

**16.117.1 Member Typedef Documentation****16.117.1.1 IntType**

```
using IntType = typename make_unsigned_int<Element>::type
```

**16.117.2 Constructor & Destructor Documentation**

### 16.117.2.1 FloatingPointTestDistribution()

```
FloatingPointTestDistribution ( ) [inline]
```

## 16.117.3 Member Function Documentation

### 16.117.3.1 operator>()()

```
Element operator() (
    Generator & g ) [inline]
```

The documentation for this class was generated from the following file:

- [test-simd.C](#)

## 16.118 ForStrategy1D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy1D](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- [ForStrategy1D](#) (const blocksize\_t b, const blocksize\_t e, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- void [build](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [begin](#) () const
- blocksize\_t [end](#) () const
- blocksize\_t [numblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [operator++](#) ()

### Protected Attributes

- blocksize\_t [ibeg](#)
- blocksize\_t [iend](#)
- blocksize\_t [current](#)
- blocksize\_t [firstBlockSize](#)
- blocksize\_t [lastBlockSize](#)
- blocksize\_t [changeBS](#)
- blocksize\_t [numBlock](#)

## 16.118.1 Constructor & Destructor Documentation

### 16.118.1.1 ForStrategy1D() [1/2]

```
ForStrategy1D (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

### 16.118.1.2 ForStrategy1D() [2/2]

```
ForStrategy1D (
    const blocksize_t b,
    const blocksize_t e,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

## 16.118.2 Member Function Documentation

### 16.118.2.1 build()

```
void build (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

### 16.118.2.2 initialize()

```
blocksize_t initialize ( ) [inline]
```

### 16.118.2.3 isTerminated()

```
bool isTerminated ( ) const [inline]
```

### 16.118.2.4 begin()

```
blocksize_t begin ( ) const [inline]
```

### 16.118.2.5 end()

```
blocksize_t end ( ) const [inline]
```

### 16.118.2.6 numblocks()

```
blocksize_t numblocks ( ) const [inline]
```

### 16.118.2.7 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

### 16.118.2.8 operator++()

```
blocksize_t operator++ ( ) [inline]
```

## 16.118.3 Field Documentation

### 16.118.3.1 ibeg

```
blocksize_t ibeg [protected]
```

### 16.118.3.2 iend

```
blocksize_t iend [protected]
```

**16.118.3.3 current**

```
blocksize_t current [protected]
```

**16.118.3.4 firstBlockSize**

```
blocksize_t firstBlockSize [protected]
```

**16.118.3.5 lastBlockSize**

```
blocksize_t lastBlockSize [protected]
```

**16.118.3.6 changeBS**

```
blocksize_t changeBS [protected]
```

**16.118.3.7 numBlock**

```
blocksize_t numBlock [protected]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.119 ForStrategy2D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy2D](#) (const blocksize\_t m, const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [ibegin](#) () const
- blocksize\_t [jbegin](#) () const
- blocksize\_t [iend](#) () const
- blocksize\_t [jend](#) () const
- blocksize\_t [operator++](#) ()
- blocksize\_t [rownumblocks](#) () const
- blocksize\_t [colnumblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [rowblockindex](#) () const
- blocksize\_t [colblockindex](#) () const

### Protected Attributes

- blocksize\_t [\\_ibeg](#)
- blocksize\_t [\\_iend](#)
- blocksize\_t [\\_jbeg](#)
- blocksize\_t [\\_jend](#)
- blocksize\_t [rowBlockSize](#)
- blocksize\_t [colBlockSize](#)
- blocksize\_t [current](#)
- blocksize\_t [lastRBS](#)
- blocksize\_t [lastCBS](#)

- blocksize\_t [changeRBS](#)
- blocksize\_t [changeCBS](#)
- blocksize\_t [numRowBlock](#)
- blocksize\_t [numColBlock](#)
- blocksize\_t [BLOCKS](#)

## Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [ForStrategy2D](#) &FS2D)

## 16.119.1 Constructor & Destructor Documentation

### 16.119.1.1 ForStrategy2D()

```
ForStrategy2D (
    const blocksize_t m,
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

## 16.119.2 Member Function Documentation

### 16.119.2.1 initialize()

```
blocksize_t initialize ( ) [inline]
```

### 16.119.2.2 isTerminated()

```
bool isTerminated ( ) const [inline]
```

### 16.119.2.3 ibegin()

```
blocksize_t ibegin ( ) const [inline]
```

### 16.119.2.4 jbegin()

```
blocksize_t jbegin ( ) const [inline]
```

### 16.119.2.5 iend()

```
blocksize_t iend ( ) const [inline]
```

### 16.119.2.6 jend()

```
blocksize_t jend ( ) const [inline]
```

### 16.119.2.7 operator++()

```
blocksize_t operator++ ( ) [inline]
```

#### 16.119.2.8 rownumblocks()

```
blocksize_t rownumblocks ( ) const [inline]
```

#### 16.119.2.9 colnumblocks()

```
blocksize_t colnumblocks ( ) const [inline]
```

#### 16.119.2.10 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

#### 16.119.2.11 rowblockindex()

```
blocksize_t rowblockindex ( ) const [inline]
```

#### 16.119.2.12 colblockindex()

```
blocksize_t colblockindex ( ) const [inline]
```

### 16.119.3 Friends And Related Function Documentation

#### 16.119.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const ForStrategy2D< blocksize_t, Cut, Param > & FS2D ) [friend]
```

### 16.119.4 Field Documentation

#### 16.119.4.1 \_ibeg

```
blocksize_t _ibeg [protected]
```

#### 16.119.4.2 \_iend

```
blocksize_t _iend [protected]
```

#### 16.119.4.3 \_jbeg

```
blocksize_t _jbeg [protected]
```

#### 16.119.4.4 \_jend

```
blocksize_t _jend [protected]
```

**16.119.4.5 rowBlockSize**

`blocksize_t rowBlockSize` [protected]

**16.119.4.6 colBlockSize**

`blocksize_t colBlockSize` [protected]

**16.119.4.7 current**

`blocksize_t current` [protected]

**16.119.4.8 lastRBS**

`blocksize_t lastRBS` [protected]

**16.119.4.9 lastCBS**

`blocksize_t lastCBS` [protected]

**16.119.4.10 changeRBS**

`blocksize_t changeRBS` [protected]

**16.119.4.11 changeCBS**

`blocksize_t changeCBS` [protected]

**16.119.4.12 numRowsBlock**

`blocksize_t numRowsBlock` [protected]

**16.119.4.13 numColBlock**

`blocksize_t numColBlock` [protected]

**16.119.4.14 BLOCKS**

`blocksize_t BLOCKS` [protected]

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.120 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.121 `ftrmmLeftLowerNoTransUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.122 `ftrmmLeftLowerTransNonUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.123 `ftrmmLeftLowerTransUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.124 `ftrmmLeftUpperNoTransNonUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.125 `ftrmmLeftUpperNoTransUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.126 `ftrmmLeftUpperTransNonUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.127 `ftrmmLeftUpperTransUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.128 `ftrmmRightLowerNoTransNonUnit`< `Element` > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.129 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.130 ftrmmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.131 ftrmmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.132 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.133 ftrmmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.134 ftrmmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.135 ftrmmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.136 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.137 `ftrsmLeftLowerNoTransUnit< Element > Class Template Reference`

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.138 `ftrsmLeftLowerTransNonUnit< Element > Class Template Reference`

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.139 `ftrsmLeftLowerTransUnit< Element > Class Template Reference`

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.140 `ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference`

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

### 16.140.1 Detailed Description

```
template<class Element>
```

```
class FFLAS::Protected::ftrsmLeftUpperNoTransNonUnit< Element >
```

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

Compute the maximal dimension  $k$ , such that a unit diagonal triangular system of dimension  $k$  can be solved over  $Z$  without overflow of the underlying floating point representation.

**Bibliography** • Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Parameters

$F$	Finite Field/Ring of the computation
-----	--------------------------------------

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.141 `ftrsmLeftUpperNoTransUnit< Element > Class Template Reference`

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.142 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.143 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.144 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.145 ftrsmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.146 ftrsmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.147 ftrsmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.148 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.149 ftrsmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.150 `ftrsmRightUpperTransNonUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.151 `ftrsmRightUpperTransUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.152 GenericTag Struct Reference

default is generic

```
#include <field-traits.h>
```

### 16.152.1 Detailed Description

default is generic

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.153 GenericTag Struct Reference

generic ring.

```
#include <field-traits.h>
```

### 16.153.1 Detailed Description

generic ring.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.154 Grain Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.155 `has_minus_eq_impl< C >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.155.1 Field Documentation

#### 16.155.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.156 has\_minus\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

#### 16.156.1 Field Documentation

##### 16.156.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.157 has\_mul\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

#### 16.157.1 Field Documentation

##### 16.157.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.158 has\_mul\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

#### 16.158.1 Field Documentation

**16.158.1.1 value**

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.159 has\_operation< T > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

**Static Public Attributes**

- static constexpr bool [value](#)

**16.159.1 Field Documentation****16.159.1.1 value**

```
constexpr bool value [static], [constexpr]
```

**Initial value:**

```
= (has_plus<T>::value && has_minus<T>::value && has_equal<T>::value &&
    has_plus_eq<T>::value && has_minus_eq<T>::value && has_mul<T>::value
    && has_mul_eq<T>::value)
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.160 has\_plus\_eq\_impl< C > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

**Static Public Attributes**

- static constexpr bool [value](#) = type::value

**16.160.1 Field Documentation****16.160.1.1 value**

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.161 has\_plus\_impl< C > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

**Static Public Attributes**

- static constexpr bool [value](#) = type::value

### 16.161.1 Field Documentation

#### 16.161.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.162 HelperFlag Struct Reference

```
#include <fflas_sparse.h>
```

### Static Public Attributes

- static constexpr uint64\_t [none](#) = 0\_ui64
- static constexpr uint64\_t [coo](#) = 1\_ui64
- static constexpr uint64\_t [csr](#) = 1\_ui64 << 1
- static constexpr uint64\_t [ell](#) = 1\_ui64 << 2
- static constexpr uint64\_t [aut](#) = 1\_ui64 << 32
- static constexpr uint64\_t [pm1](#) = 1\_ui64 << 33

### 16.162.1 Field Documentation

#### 16.162.1.1 none

```
constexpr uint64_t none = 0_ui64 [static], [constexpr]
```

#### 16.162.1.2 coo

```
constexpr uint64_t coo = 1_ui64 [static], [constexpr]
```

#### 16.162.1.3 csr

```
constexpr uint64_t csr = 1_ui64 << 1 [static], [constexpr]
```

#### 16.162.1.4 ell

```
constexpr uint64_t ell = 1_ui64 << 2 [static], [constexpr]
```

#### 16.162.1.5 aut

```
constexpr uint64_t aut = 1_ui64 << 32 [static], [constexpr]
```

#### 16.162.1.6 pm1

```
constexpr uint64_t pm1 = 1_ui64 << 33 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.163 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.164 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

### Data Fields

- [Field::Element](#) p
- double [invp](#)
- double [min](#)
- double [max](#)
- int64\_t [pow50rem](#)

## 16.164.1 Constructor & Destructor Documentation

### 16.164.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

### 16.164.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

## 16.164.2 Field Documentation

### 16.164.2.1 p

```
Field::Element p
```

### 16.164.2.2 invp

```
double invp
```

### 16.164.2.3 min

```
double min
```

### 16.164.2.4 max

```
double max
```

### 16.164.2.5 pow50rem

```
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.165 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

### Data Fields

- [Field::Element](#) p

### 16.165.1 Constructor & Destructor Documentation

#### 16.165.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

#### 16.165.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

### 16.165.2 Field Documentation

#### 16.165.2.1 p

```
Field::Element p
```

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.166 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

### Data Fields

- [Field::Element](#) p

## 16.166.1 Constructor & Destructor Documentation

### 16.166.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

### 16.166.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

## 16.166.2 Field Documentation

### 16.166.2.1 p

```
Field::Element p
```

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.167 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference

### Public Member Functions

- [HelperMod \(\)](#)
- [HelperMod \(const Field &F\)](#)

### Data Fields

- [Field::Element p](#)
- [Field::Element invp](#)
- [Field::Element min](#)
- [Field::Element max](#)

## 16.167.1 Constructor & Destructor Documentation

### 16.167.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

### 16.167.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

## 16.167.2 Field Documentation

### 16.167.2.1 p

`Field::Element` p

### 16.167.2.2 invp

`Field::Element` invp

### 16.167.2.3 min

`Field::Element` min

### 16.167.2.4 max

`Field::Element` max

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.168 Hybrid Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.169 Info Struct Reference

### Public Member Functions

- [Info](#) (uint64\_t it, uint64\_t s, uint64\_t p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

### Data Fields

- uint64\_t [size](#) = 0
- uint64\_t [perm](#) = 0
- uint64\_t [begin](#) = 0

### 16.169.1 Constructor & Destructor Documentation

#### 16.169.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

**16.169.1.2 Info()** [2/4]

```
Info ( ) [default]
```

**16.169.1.3 Info()** [3/4]

```
Info (
    const Info & ) [default]
```

**16.169.1.4 Info()** [4/4]

```
Info (
    Info && ) [default]
```

**16.169.2 Member Function Documentation****16.169.2.1 operator=()** [1/2]

```
Info& operator= (
    const Info & ) [default]
```

**16.169.2.2 operator=()** [2/2]

```
Info& operator= (
    Info && ) [default]
```

**16.169.3 Field Documentation****16.169.3.1 size**

```
uint64_t size = 0
```

**16.169.3.2 perm**

```
uint64_t perm = 0
```

**16.169.3.3 begin**

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

**16.170 Info Struct Reference****Public Member Functions**

- [Info](#) (uint64\_t it, uint64\_t s, uint64\_t p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default

- `Info (Info &&)=default`
- `Info & operator= (const Info &)=default`
- `Info & operator= (Info &&)=default`

## Data Fields

- `uint64_t size = 0`
- `uint64_t perm = 0`
- `uint64_t begin = 0`

## 16.170.1 Constructor & Destructor Documentation

### 16.170.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

### 16.170.1.2 Info() [2/4]

```
Info ( ) [default]
```

### 16.170.1.3 Info() [3/4]

```
Info (
    const Info & ) [default]
```

### 16.170.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

## 16.170.2 Member Function Documentation

### 16.170.2.1 operator=() [1/2]

```
Info& operator= (
    const Info & ) [default]
```

### 16.170.2.2 operator=() [2/2]

```
Info& operator= (
    Info && ) [default]
```

## 16.170.3 Field Documentation

**16.170.3.1 size**

```
uint64_t size = 0
```

**16.170.3.2 perm**

```
uint64_t perm = 0
```

**16.170.3.3 begin**

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

**16.171 is\_all\_same< Args > Struct Template Reference**

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

**16.172 is\_all\_same< T, Args... > Struct Template Reference****Static Public Attributes**

- static constexpr bool [value](#) = [ALL](#)<std::is\_same<typename decay<T>::type, typename decay<Args>↵::type>::value...>::value

**16.172.1 Field Documentation****16.172.1.1 value**

```
constexpr bool value = ALL<std::is_same<typename decay<T>::type, typename decay<Args>↵::type>::value...>::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

**16.173 is\_all\_same<> Struct Reference****Static Public Attributes**

- static constexpr bool [value](#) = true

**16.173.1 Field Documentation****16.173.1.1 value**

```
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.174 `is_simd< T >` Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using `type` = `std::integral_constant< bool, false >`

### Static Public Attributes

- static constexpr const bool `value` = false

### 16.174.1 Member Typedef Documentation

#### 16.174.1.1 `type`

```
using type = std::integral_constant<bool, false>
```

### 16.174.2 Field Documentation

#### 16.174.2.1 `value`

```
constexpr const bool value = false [static], [constexpr]
```

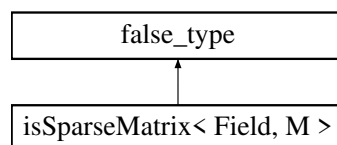
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.175 `isSparseMatrix< Field, M >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, M >`:



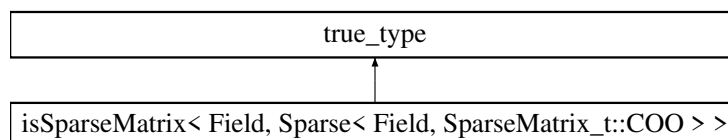
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.176 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >`:



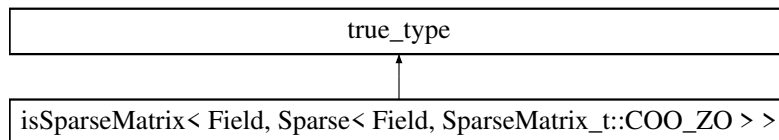
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.177 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >:



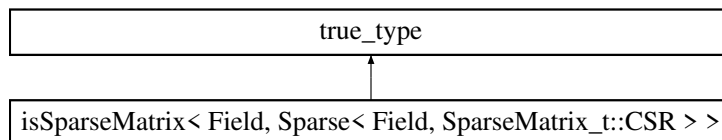
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.178 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >:



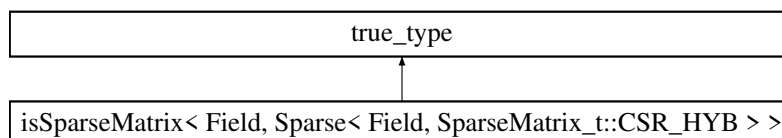
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.179 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >:



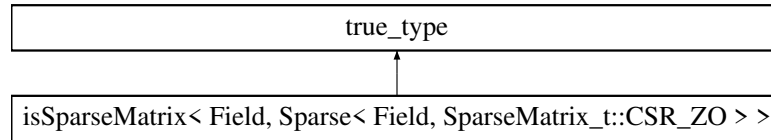
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.180 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



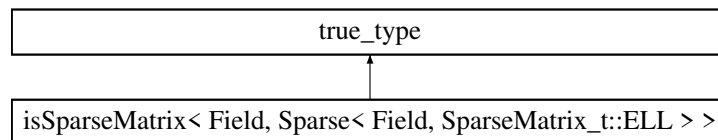
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.181 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >:



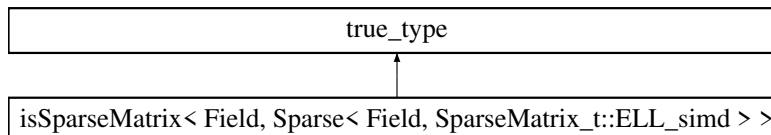
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.182 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >:



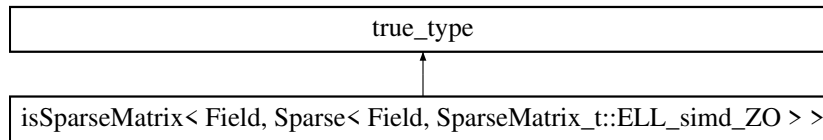
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.183 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



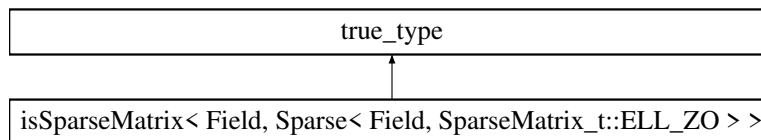
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.184 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



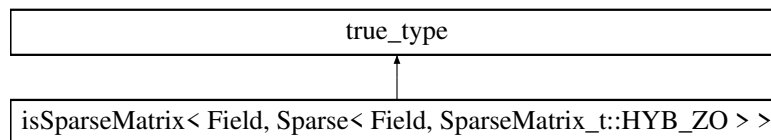
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.185 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >:



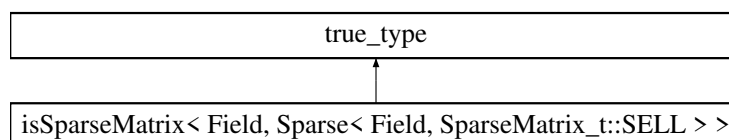
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.186 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >:



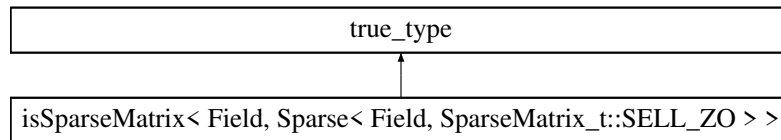
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.187 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



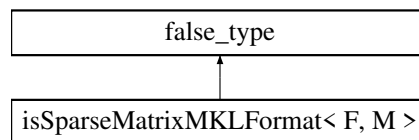
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.188 isSparseMatrixMKLFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



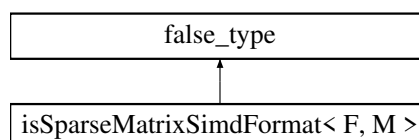
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.189 isSparseMatrixSimdFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



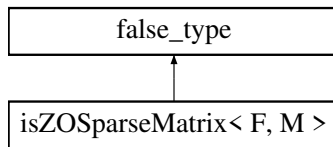
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.190 isZOSparseMatrix< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< F, M >:



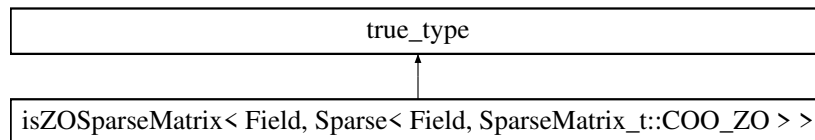
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.191 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >:



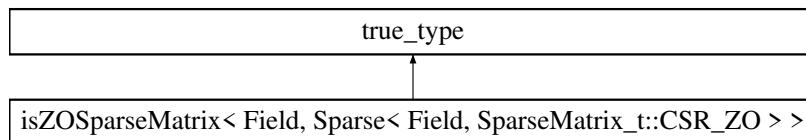
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.192 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



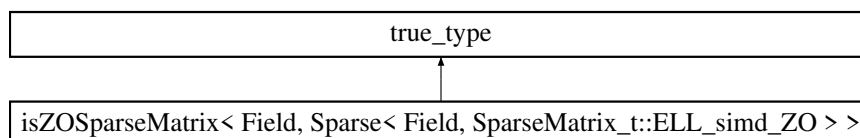
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.193 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



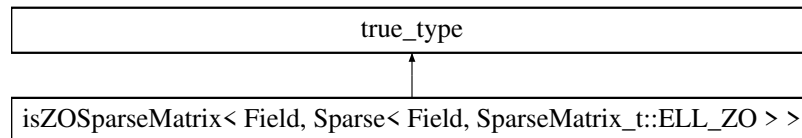
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.194 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



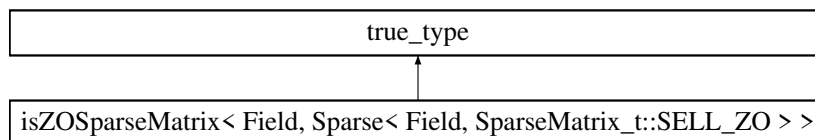
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.195 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.196 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.197 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.

```
#include <field-traits.h>
```

### 16.197.1 Detailed Description

Performs field operations with delayed mod only when necessary. Result may not be reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.198 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.199 limits< char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef char [T](#)

### Static Public Member Functions

- constexpr static char [max](#) () noexcept
- constexpr static char [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.199.1 Member Typedef Documentation

#### 16.199.1.1 T

```
typedef char T
```

### 16.199.2 Member Function Documentation

#### 16.199.2.1 max()

```
constexpr static char max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.199.2.2 min()

```
constexpr static char min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.199.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.200 limits< double > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef double [T](#)

## Static Public Member Functions

- constexpr static int64\_t [max](#) () noexcept
- constexpr static int64\_t [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.200.1 Member Typedef Documentation

#### 16.200.1.1 T

```
typedef double T
```

### 16.200.2 Member Function Documentation

#### 16.200.2.1 max()

```
constexpr static int64_t max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.200.2.2 min()

```
constexpr static int64_t min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.200.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.201 limits< float > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef float [T](#)

### Static Public Member Functions

- constexpr static int32\_t [max](#) () noexcept
- constexpr static int32\_t [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.201.1 Member Typedef Documentation

#### 16.201.1.1 T

```
typedef float T
```

## 16.201.2 Member Function Documentation

### 16.201.2.1 max()

```
constexpr static int32_t max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.201.2.2 min()

```
constexpr static int32_t min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.201.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.202 limits< Givaro::Integer > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef Givaro::Integer [T](#)

### Static Public Member Functions

- constexpr static int [max](#) () noexcept
- constexpr static int [min](#) () noexcept

## 16.202.1 Member Typedef Documentation

### 16.202.1.1 T

```
typedef Givaro::Integer T
```

## 16.202.2 Member Function Documentation

### 16.202.2.1 max()

```
constexpr static int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.202.2.2 min()

```
constexpr static int min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.203 limits< int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef int [T](#)

### Static Public Member Functions

- constexpr static int [max](#) () noexcept
- constexpr static int [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.203.1 Member Typedef Documentation

#### 16.203.1.1 T

```
typedef int T
```

### 16.203.2 Member Function Documentation

#### 16.203.2.1 max()

```
constexpr static int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.203.2.2 min()

```
constexpr static int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.203.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.204 limits< long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef long [T](#)

### Static Public Member Functions

- constexpr static long [max](#) () noexcept
- constexpr static long [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

## 16.204.1 Member Typedef Documentation

### 16.204.1.1 T

`typedef long T`

## 16.204.2 Member Function Documentation

### 16.204.2.1 max()

`constexpr static long max ( ) [inline], [static], [constexpr], [noexcept]`

### 16.204.2.2 min()

`constexpr static long min ( ) [inline], [static], [constexpr], [noexcept]`

### 16.204.2.3 digits()

`constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.205 limits< long long > Struct Reference

`#include <flimits.h>`

### Public Types

- `typedef long long T`

### Static Public Member Functions

- `constexpr static long long max () noexcept`
- `constexpr static long long min () noexcept`
- `constexpr static int32_t digits () noexcept`

## 16.205.1 Member Typedef Documentation

### 16.205.1.1 T

`typedef long long T`

## 16.205.2 Member Function Documentation

### 16.205.2.1 max()

`constexpr static long long max ( ) [inline], [static], [constexpr], [noexcept]`

**16.205.2.2 min()**

```
constexpr static long long min ( ) [inline], [static], [constexpr], [noexcept]
```

**16.205.2.3 digits()**

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

**16.206 limits< RecInt::rint< K > > Struct Template Reference**

```
#include <flimits.h>
```

**Public Types**

- typedef [RecInt::ruint](#)< K > [T](#)

**Static Public Member Functions**

- constexpr static [RecInt::rint](#)< K > [max](#) ( ) noexcept
- constexpr static [RecInt::rint](#)< K > [min](#) ( ) noexcept

**16.206.1 Member Typedef Documentation****16.206.1.1 T**

```
typedef RecInt::ruint<K> T
```

**16.206.2 Member Function Documentation****16.206.2.1 max()**

```
constexpr static RecInt::rint<K> max ( ) [inline], [static], [constexpr], [noexcept]
```

**16.206.2.2 min()**

```
constexpr static RecInt::rint<K> min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

**16.207 limits< RecInt::ruint< K > > Struct Template Reference**

```
#include <flimits.h>
```

**Public Types**

- typedef [RecInt::ruint](#)< K > [T](#)

## Static Public Member Functions

- constexpr static [RecInt::ruint](#)< K > [max](#) () noexcept
- constexpr static [RecInt::ruint](#)< K > [min](#) () noexcept

### 16.207.1 Member Typedef Documentation

#### 16.207.1.1 T

```
typedef RecInt::ruint<K> T
```

### 16.207.2 Member Function Documentation

#### 16.207.2.1 max()

```
constexpr static RecInt::ruint<K> max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.207.2.2 min()

```
constexpr static RecInt::ruint<K> min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.208 limits< short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef short int [T](#)

### Static Public Member Functions

- constexpr static short int [max](#) () noexcept
- constexpr static short int [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.208.1 Member Typedef Documentation

#### 16.208.1.1 T

```
typedef short int T
```

### 16.208.2 Member Function Documentation

#### 16.208.2.1 max()

```
constexpr static short int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.208.2.2 min()

```
constexpr static short int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.208.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.209 limits< signed char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef signed char [T](#)

### Static Public Member Functions

- constexpr static signed char [max](#) () noexcept
- constexpr static signed char [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.209.1 Member Typedef Documentation

#### 16.209.1.1 T

```
typedef signed char T
```

### 16.209.2 Member Function Documentation

#### 16.209.2.1 max()

```
constexpr static signed char max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.209.2.2 min()

```
constexpr static signed char min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.209.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.210 limits< unsigned char > Struct Reference

```
#include <flimits.h>
```

## Public Types

- typedef unsigned char [T](#)

## Static Public Member Functions

- constexpr static unsigned char [max](#) () noexcept
- constexpr static unsigned char [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.210.1 Member Typedef Documentation

#### 16.210.1.1 T

typedef unsigned char [T](#)

### 16.210.2 Member Function Documentation

#### 16.210.2.1 max()

constexpr static unsigned char max ( ) [inline], [static], [constexpr], [noexcept]

#### 16.210.2.2 min()

constexpr static unsigned char min ( ) [inline], [static], [constexpr], [noexcept]

#### 16.210.2.3 digits()

constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.211 limits< unsigned int > Struct Reference

```
#include <flimits.h>
```

## Public Types

- typedef unsigned int [T](#)

## Static Public Member Functions

- constexpr static unsigned int [max](#) () noexcept
- constexpr static unsigned int [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.211.1 Member Typedef Documentation

### 16.211.1.1 T

```
typedef unsigned int T
```

## 16.211.2 Member Function Documentation

### 16.211.2.1 max()

```
constexpr static unsigned int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.211.2.2 min()

```
constexpr static unsigned int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.211.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.212 limits< unsigned long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned long [T](#)

### Static Public Member Functions

- constexpr static unsigned long [max](#) ( ) noexcept
- constexpr static unsigned long [min](#) ( ) noexcept
- constexpr static int32\_t [digits](#) ( ) noexcept

## 16.212.1 Member Typedef Documentation

### 16.212.1.1 T

```
typedef unsigned long T
```

## 16.212.2 Member Function Documentation

### 16.212.2.1 max()

```
constexpr static unsigned long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.212.2.2 min()

```
constexpr static unsigned long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.212.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.213 limits< unsigned long long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned long long [T](#)

### Static Public Member Functions

- constexpr static unsigned long long [max](#) () noexcept
- constexpr static unsigned long long [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.213.1 Member Typedef Documentation

#### 16.213.1.1 T

```
typedef unsigned long long T
```

### 16.213.2 Member Function Documentation

#### 16.213.2.1 max()

```
constexpr static unsigned long long max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.213.2.2 min()

```
constexpr static unsigned long long min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.213.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.214 limits< unsigned short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned short int [T](#)

## Static Public Member Functions

- constexpr static unsigned short int [max](#) () noexcept
- constexpr static unsigned short int [min](#) () noexcept
- constexpr static int32\_t [digits](#) () noexcept

### 16.214.1 Member Typedef Documentation

#### 16.214.1.1 T

```
typedef unsigned short int T
```

### 16.214.2 Member Function Documentation

#### 16.214.2.1 max()

```
constexpr static unsigned short int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.214.2.2 min()

```
constexpr static unsigned short int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.214.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.215 MachineFloatTag Struct Reference

float or double

```
#include <field-traits.h>
```

### 16.215.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.216 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

```
#include <field-traits.h>
```

### 16.216.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.217 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) > [Self\\_t](#)
- typedef [associatedDelayedField](#)< const [Field](#) >::type [DelayedField\\_t](#)
- typedef [associatedDelayedField](#)< const [Field](#) >::field [DelayedField](#)
- typedef [DelayedField::Element](#) [DFElt](#)

### Public Member Functions

- void [initC](#) ()
- void [initA](#) ()
- void [initB](#) ()
- void [initOut](#) ()
- [size\\_t](#) [MaxDelayedDim](#) ([DFElt](#) beta)
- bool [Aunfit](#) ()
- bool [Bunfit](#) ()
- void [setOutBounds](#) (const [size\\_t](#) k, const [DFElt](#) alpha, const [DFElt](#) beta)
- bool [checkA](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda)
- bool [checkB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) B, const [size\\_t](#) ldb)
- bool [checkOut](#) (const [Field](#) &F, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda)
- bool [checkOut](#) (const [Field](#) &F, [FFLAS\\_UPLO](#) uplo, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda)
- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, [size\\_t](#) m, [size\\_t](#) k, [size\\_t](#) n, [ParSeqTrait\\_PS](#))
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait\\_PS](#)=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- [MMHelper](#) (const [Field](#) &F, int w, [DFElt\\_Amin](#), [DFElt\\_Amax](#), [DFElt\\_Bmin](#), [DFElt\\_Bmax](#), [DFElt\\_Cmin](#), [DFElt\\_Cmax](#), [ParSeqTrait\\_PS](#)=[ParSeqTrait](#)())

### Data Fields

- int [recLevel](#)
- [DFElt](#) [FieldMin](#)
- [DFElt](#) [FieldMax](#)
- [DFElt](#) [Amin](#)
- [DFElt](#) [Amax](#)
- [DFElt](#) [Bmin](#)
- [DFElt](#) [Bmax](#)
- [DFElt](#) [Cmin](#)
- [DFElt](#) [Cmax](#)
- [DFElt](#) [Outmin](#)
- [DFElt](#) [Outmax](#)
- [DFElt](#) [MaxStorableValue](#)
- const [DelayedField\\_t](#) [delayedField](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.217.1 Member Typedef Documentation

### 16.217.1.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

### 16.217.1.2 DelayedField\_t

```
typedef associatedDelayedField<const Field>::type DelayedField_t
```

### 16.217.1.3 DelayedField

```
typedef associatedDelayedField<const Field>::field DelayedField
```

### 16.217.1.4 DFElt

```
typedef DelayedField::Element DFElt
```

## 16.217.2 Constructor & Destructor Documentation

### 16.217.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

### 16.217.2.2 MMHelper() [2/5]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

### 16.217.2.3 MMHelper() [3/5]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

### 16.217.2.4 MMHelper() [4/5]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

**16.217.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int w,
    DFelt _Amin,
    DFelt _Amax,
    DFelt _Bmin,
    DFelt _Bmax,
    DFelt _Cmin,
    DFelt _Cmax,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.217.3 Member Function Documentation****16.217.3.1 initC()**

```
void initC ( ) [inline]
```

**16.217.3.2 initA()**

```
void initA ( ) [inline]
```

**16.217.3.3 initB()**

```
void initB ( ) [inline]
```

**16.217.3.4 initOut()**

```
void initOut ( ) [inline]
```

**16.217.3.5 MaxDelayedDim()**

```
size_t MaxDelayedDim (
    DFelt beta ) [inline]
```

**16.217.3.6 Aunfit()**

```
bool Aunfit ( ) [inline]
```

**16.217.3.7 Bunfit()**

```
bool Bunfit ( ) [inline]
```

**16.217.3.8 setOutBounds()**

```
void setOutBounds (
    const size_t k,
    const DFelt alpha,
    const DFelt beta ) [inline]
```

**16.217.3.9 checkA()**

```
bool checkA (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.217.3.10 checkB()**

```
bool checkB (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

**16.217.3.11 checkOut() [1/2]**

```
bool checkOut (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.217.3.12 checkOut() [2/2]**

```
bool checkOut (
    const Field & F,
    FFLAS_UPLO uplo,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.217.4 Friends And Related Function Documentation****16.217.4.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.217.5 Field Documentation****16.217.5.1 recLevel**

```
int recLevel
```

#### 16.217.5.2 FieldMin

`DFElt` FieldMin

#### 16.217.5.3 FieldMax

`DFElt` FieldMax

#### 16.217.5.4 Amin

`DFElt` Amin

#### 16.217.5.5 Amax

`DFElt` Amax

#### 16.217.5.6 Bmin

`DFElt` Bmin

#### 16.217.5.7 Bmax

`DFElt` Bmax

#### 16.217.5.8 Cmin

`DFElt` Cmin

#### 16.217.5.9 Cmax

`DFElt` Cmax

#### 16.217.5.10 Outmin

`DFElt` Outmin

#### 16.217.5.11 Outmax

`DFElt` Outmax

#### 16.217.5.12 MaxStorableValue

`DFElt` MaxStorableValue

#### 16.217.5.13 delayedField

const `DelayedField_t` delayedField

#### 16.217.5.14 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.218 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [FFPACK::RNSInteger](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer [Amax](#), Givaro::Integer [Bmax](#))
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, size\_t m, size\_t n, size\_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2 , class A2 , class M2 , class PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.218.1 Member Typedef Documentation

### 16.218.1.1 Self\_t

```
typedef MMHelper<FFPACK::RNSInteger<E>, AlgoTrait,ModeCategories::DefaultTag, ParSeqTrait>
Self\_t
```

## 16.218.2 Constructor & Destructor Documentation

### 16.218.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

**16.218.2.2 MMHelper() [2/5]**

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

**16.218.2.3 MMHelper() [3/5]**

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.218.2.4 MMHelper() [4/5]**

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.218.2.5 MMHelper() [5/5]**

```
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

**16.218.3 Member Function Documentation****16.218.3.1 setNorm()**

```
void setNorm (
    Givaro::Integer p ) [inline]
```

**16.218.4 Friends And Related Function Documentation****16.218.4.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.218.5 Field Documentation****16.218.5.1 normA**

```
Givaro::Integer normA
```

### 16.218.5.2 normB

Givaro::Integer normB

### 16.218.5.3 recLevel

int recLevel

### 16.218.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.219 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer [Amax](#), Givaro::Integer [Bmax](#))
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, size\_t m, size\_t n, size\_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)()
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 > [MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.219.1 Member Typedef Documentation

### 16.219.1.1 Self\_t

typedef [MMHelper](#)<[FFPACK::RNSIntegerMod](#)<E>, [AlgoTrait](#),[ModeCategories::DefaultTag](#), [ParSeqTrait](#)> [Self\\_t](#)

## 16.219.2 Constructor & Destructor Documentation

### 16.219.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

### 16.219.2.2 MMHelper() [2/5]

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

### 16.219.2.3 MMHelper() [3/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.219.2.4 MMHelper() [4/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.219.2.5 MMHelper() [5/5]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

## 16.219.3 Member Function Documentation

### 16.219.3.1 setNorm()

```
void setNorm (
    Givaro::Integer p ) [inline]
```

## 16.219.4 Friends And Related Function Documentation

### 16.219.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.219.5 Field Documentation

### 16.219.5.1 normA

Givaro::Integer normA

### 16.219.5.2 normB

Givaro::Integer normB

### 16.219.5.3 recLevel

int recLevel

### 16.219.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.220 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< Dest >, [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.220.1 Member Typedef Documentation

### 16.220.1.1 Self\_t

typedef [MMHelper](#)<[Field](#),[AlgoTrait](#), [ModeCategories::ConvertTo](#)<Dest>,[ParSeqTrait](#)> [Self\\_t](#)

## 16.220.2 Constructor & Destructor Documentation

### 16.220.2.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

### 16.220.2.2 MMHelper() [2/4]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

### 16.220.2.3 MMHelper() [3/4]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

### 16.220.2.4 MMHelper() [4/4]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

## 16.220.3 Friends And Related Function Documentation

### 16.220.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.220.4 Field Documentation

### 16.220.4.1 recLevel

```
int recLevel
```

### 16.220.4.2 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.221 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Self\_t

### Public Member Functions

- MMHelper ()
- template<class F2 , class A2 , class M2 , class PS2 > MMHelper (MMHelper< F2, A2, M2, PS2 > H2)
- MMHelper (Givaro::Integer Amax, Givaro::Integer Bmax)
- MMHelper (const Field &F, size\_t m, size\_t n, size\_t k, ParSeqTrait PS=ParSeqTrait())
- MMHelper (const Field &F, int wino, ParSeqTrait PS=ParSeqTrait())
- void setNorm (Givaro::Integer p)

### Data Fields

- Givaro::Integer normA
- Givaro::Integer normB
- int recLevel
- ParSeqTrait parseq

### Friends

- std::ostream & operator<< (std::ostream &out, const Self\_t &M)

## 16.221.1 Member Typedef Documentation

### 16.221.1.1 Self\_t

```
typedef MMHelper<Field, AlgoTrait,ModeCategories::ConvertTo<ElementCategories::RNSElementTag>, ParSeqTrait> Self_t
```

## 16.221.2 Constructor & Destructor Documentation

### 16.221.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

### 16.221.2.2 MMHelper() [2/5]

```
MMHelper ( MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

**16.221.2.3 MMHelper() [3/5]**

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

**16.221.2.4 MMHelper() [4/5]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.221.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.221.3 Member Function Documentation****16.221.3.1 setNorm()**

```
void setNorm (
    Givaro::Integer p ) [inline]
```

**16.221.4 Friends And Related Function Documentation****16.221.4.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.221.5 Field Documentation****16.221.5.1 normA**

```
Givaro::Integer normA
```

**16.221.5.2 normB**

```
Givaro::Integer normB
```

### 16.221.5.3 recLevel

```
int recLevel
```

### 16.221.5.4 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.222 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

FGEMM Helper for Default and ConvertTo modes of operation.

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 > [MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

### 16.222.1 Detailed Description

```
template<class Field, typename AlgoTrait, typename ParSeqTrait>
struct FFLAS::MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
```

FGEMM Helper for Default and ConvertTo modes of operation.

### 16.222.2 Member Typedef Documentation

#### 16.222.2.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::DefaultTag,ParSeqTrait> Self\_t
```

### 16.222.3 Constructor & Destructor Documentation

**16.222.3.1 MMHelper() [1/4]**

```
MMHelper ( ) [inline]
```

**16.222.3.2 MMHelper() [2/4]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

**16.222.3.3 MMHelper() [3/4]**

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.222.3.4 MMHelper() [4/4]**

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

**16.222.4 Friends And Related Function Documentation****16.222.4.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.222.5 Field Documentation****16.222.5.1 recLevel**

```
int recLevel
```

**16.222.5.2 parseq**

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.223 ModeTraits< Field > Struct Template Reference**

```
ModeTraits.
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::DefaultTag](#) value

### 16.223.1 Detailed Description

```
template<class Field>
struct FFLAS::ModeTraits< Field >
```

[ModeTraits](#).

### 16.223.2 Member Typedef Documentation

#### 16.223.2.1 value

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.224 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::DelayedTag](#) value

### 16.224.1 Member Typedef Documentation

#### 16.224.1.1 value

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.225 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.225.1 Member Typedef Documentation

**16.225.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.226 ModeTraits< Givaro::Modular< int16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

**16.226.1 Member Typedef Documentation****16.226.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.227 ModeTraits< Givaro::Modular< int32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

**16.227.1 Member Typedef Documentation****16.227.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.228 ModeTraits< Givaro::Modular< int64\_t, uint64\_t > > Struct Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::DefaultTag value](#)

### 16.228.1 Member Typedef Documentation

#### 16.228.1.1 value

typedef [ModeCategories::DefaultTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.229 ModeTraits< Givaro::Modular< int8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.229.1 Member Typedef Documentation

#### 16.229.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.230 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.230.1 Member Typedef Documentation

#### 16.230.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.231 ModeTraits< Givaro::Modular< uint16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.231.1 Member Typedef Documentation

#### 16.231.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.232 ModeTraits< Givaro::Modular< uint32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.232.1 Member Typedef Documentation

#### 16.232.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.233 ModeTraits< Givaro::Modular< uint8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.233.1 Member Typedef Documentation

#### 16.233.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.234 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DelayedTag](#) value

### 16.234.1 Member Typedef Documentation

#### 16.234.1.1 value

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.235 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.235.1 Member Typedef Documentation

#### 16.235.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.236 ModeTraits< Givaro::ModularBalanced< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.236.1 Member Typedef Documentation

**16.236.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.237 ModeTraits< Givaro::ModularBalanced< int32\_t > > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

**16.237.1 Member Typedef Documentation****16.237.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.238 ModeTraits< Givaro::ModularBalanced< int8\_t > > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

**16.238.1 Member Typedef Documentation****16.238.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.239 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::DefaultBoundedTag value](#)

### 16.239.1 Member Typedef Documentation

#### 16.239.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.240 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.240.1 Member Typedef Documentation

#### 16.240.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.241 ModeTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.241.1 Member Typedef Documentation

#### 16.241.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.242 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.242.1 Member Typedef Documentation

#### 16.242.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

### 16.243 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

### 16.244 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
`#include <field-traits.h>`

#### 16.244.1 Detailed Description

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
 The documentation for this struct was generated from the following file:

- [field-traits.h](#)

### 16.245 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

### 16.246 need\_field\_characteristic< Field > Struct Template Reference

#### Static Public Attributes

- static constexpr bool [value](#) = false

#### 16.246.1 Field Documentation

##### 16.246.1.1 value

constexpr bool value = false [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

### 16.247 need\_field\_characteristic< Givaro::Modular< Field > > Struct Template Reference

#### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.247.1 Field Documentation

#### 16.247.1.1 value

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.248 need\_field\_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.248.1 Field Documentation

#### 16.248.1.1 value

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.249 NoSimd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [vect\\_t](#) = T \*
- using [scalar\\_t](#) = T
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename [Field::Element](#), T >

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class TT >  
static constexpr bool [valid](#) (TT p)
- template<class TT >  
static constexpr bool [compliant](#) (TT n)

### Static Public Attributes

- static constexpr const size\_t [vect\\_size](#) = 1
- static constexpr const size\_t [alignment](#) = static\_cast<size\_t>(Alignment::Normal)

### 16.249.1 Member Typedef Documentation

**16.249.1.1 vect\_t**

```
using vect_t = T*
```

**16.249.1.2 scalar\_t**

```
using scalar_t = T
```

**16.249.1.3 aligned\_allocator**

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

**16.249.1.4 aligned\_vector**

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.249.1.5 is\_same\_element**

```
using is_same_element = std::is_same<typename Field::Element, T>
```

**16.249.2 Member Function Documentation****16.249.2.1 type\_string()**

```
static const std::string type_string ( ) [inline], [static]
```

**16.249.2.2 valid()**

```
static constexpr bool valid (
    TT p ) [inline], [static], [constexpr]
```

**16.249.2.3 compliant()**

```
static constexpr bool compliant (
    TT n ) [inline], [static], [constexpr]
```

**16.249.3 Field Documentation****16.249.3.1 vect\_size**

```
constexpr const size_t vect_size = 1 [static], [constexpr]
```

**16.249.3.2 alignment**

```
constexpr const size_t alignment = static_cast<size_t>(Alignment::Normal) [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.250 Parallel< C, P > Struct Template Reference

### Public Types

- typedef C [Cut](#)
- typedef P [Param](#)

### Public Member Functions

- [Parallel](#) (size\_t n=[NUM\\_THREADS](#))
- size\_t [numthreads](#) () const
- size\_t & [set\\_numthreads](#) (size\_t n)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Parallel](#) &p)

## 16.250.1 Member Typedef Documentation

### 16.250.1.1 Cut

```
typedef C Cut
```

### 16.250.1.2 Param

```
typedef P Param
```

## 16.250.2 Constructor & Destructor Documentation

### 16.250.2.1 Parallel()

```
Parallel (  
    size_t n = NUM\_THREADS ) [inline]
```

## 16.250.3 Member Function Documentation

### 16.250.3.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

### 16.250.3.2 set\_numthreads()

```
size_t& set_numthreads (  
    size_t n ) [inline]
```

## 16.250.4 Friends And Related Function Documentation

#### 16.250.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Parallel< C, P > & p ) [friend]
```

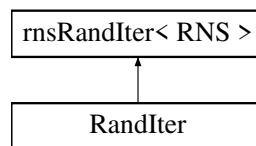
The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.251 RNSInteger< RNS >::RandIter Class Reference

```
#include <rns-integer.h>
```

Inheritance diagram for RNSInteger< RNS >::RandIter:



#### Public Member Functions

- [RandIter](#) (const [RNSInteger< RNS >](#) &F, uint64\_t seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const  
*RNS ring Element random assignement.*
- [RNS::Element random](#) () const
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- const [RNS & ring](#) () const

#### 16.251.1 Constructor & Destructor Documentation

##### 16.251.1.1 RandIter()

```
RandIter (
    const RNSInteger< RNS > & F,
    uint64_t seed = 0 ) [inline]
```

#### 16.251.2 Member Function Documentation

##### 16.251.2.1 random() [1/2]

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline], [inherited]
```

RNS ring Element random assignement.

Element is supposed to be initialized

Returns

random ring Element

**16.251.2.2 random()** [2/2]

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.251.2.3 operator>()** [1/2]

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.251.2.4 operator>()** [2/2]

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.251.2.5 ring()**

```
const RNS& ring ( ) const [inline], [inherited]
```

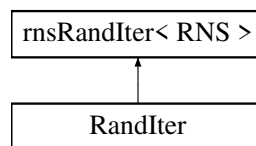
The documentation for this class was generated from the following file:

- [rns-integer.h](#)

**16.252 RNSIntegerMod< RNS >::RandIter Class Reference**

```
#include <rns-integer-mod.h>
```

Inheritance diagram for RNSIntegerMod< RNS >::RandIter:

**Public Member Functions**

- [RandIter](#) (const [RNSIntegerMod< RNS >](#) &F, uint64\_t seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element random](#) () const
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

**16.252.1 Constructor & Destructor Documentation****16.252.1.1 RandIter()**

```
RandIter (
    const RNSIntegerMod< RNS > & F,
    uint64_t seed = 0 ) [inline]
```

**16.252.2 Member Function Documentation**

**16.252.2.1 random() [1/2]**

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline]
```

**16.252.2.2 random() [2/2]**

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.252.2.3 operator>() [1/2]**

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.252.2.4 operator>() [2/2]**

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.252.2.5 ring()**

```
const RNS& ring ( ) const [inline], [inherited]
```

The documentation for this class was generated from the following file:

- [rns-integer-mod.h](#)

**16.253 readMyMachineType< Field, T > Struct Template Reference**

```
#include <read_sparse.h>
```

**Public Types**

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element\\_ptr](#) [Element\\_ptr](#)

**Public Member Functions**

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const uint64\_t dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

**16.253.1 Member Typedef Documentation****16.253.1.1 Element**

```
typedef Field::Element Element
```

**16.253.1.2 Element\_ptr**

```
typedef Field::Element\_ptr Element\_ptr
```

**16.253.2 Member Function Documentation**

### 16.253.2.1 operator>()

```
void operator() (
    const Field & F,
    Element & modulo,
    Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

## 16.254 readMyMachineType< Field, mpz\_t > Struct Template Reference

```
#include <read_sparse.h>
```

### Public Types

- typedef [Field::Element](#) Element
- typedef [Field::Element\\_ptr](#) Element\_ptr

### Public Member Functions

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const uint64\_t dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

### 16.254.1 Member Typedef Documentation

#### 16.254.1.1 Element

```
typedef Field::Element Element
```

#### 16.254.1.2 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

### 16.254.2 Member Function Documentation

#### 16.254.2.1 operator>()

```
void operator() (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

## 16.255 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.256 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.257 rint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.258 rns\_double Struct Reference

```
#include <rns-double.h>
```

### Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::Modular< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [rns\\_double](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (const [RNSIntegerMod](#)< [rns\\_double](#) > &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) (size\_t K=0)
- template<typename T >  
  void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const T \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void [init\\_transpose](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void [convert](#) (size\_t m, size\_t n, [integer](#) gamma, [integer](#) \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void [convert\\_transpose](#) (size\_t m, size\_t n, [integer](#) gamma, [integer](#) \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void [reduce](#) (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- template<size\_t K>  
  void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [Reclnt::ruint](#)< K > \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- template<size\_t K>  
  void [convert](#) (size\_t m, size\_t n, [integer](#) gamma, [Reclnt::ruint](#)< K > \*A, size\_t lda, const double \*Arns, size\_t rda, [integer](#) p=0, bool RNS\_MAJOR=false) const

## Data Fields

- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis`
- `std::vector< ModField > _field_rns`
- `integer _M`
- `std::vector< integer > _Mi`
- `std::vector< double > _MMi`
- `std::vector< double > _crt_in`
- `std::vector< double > _crt_out`
- `size_t _size`
- `size_t _pbits`
- `size_t _ldm`
- `integer _mi_sum`

## 16.258.1 Member Typedef Documentation

### 16.258.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.258.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

### 16.258.1.3 BasisElement

```
typedef double BasisElement
```

### 16.258.1.4 Element

```
typedef rns\_double\_elt Element
```

### 16.258.1.5 Element\_ptr

```
typedef rns\_double\_elt\_ptr Element\_ptr
```

### 16.258.1.6 ConstElement\_ptr

```
typedef rns\_double\_elt\_cstptr ConstElement\_ptr
```

## 16.258.2 Constructor & Destructor Documentation

**16.258.2.1 rns\_double() [1/4]**

```
rns_double (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.258.2.2 rns\_double() [2/4]**

```
rns_double (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

**16.258.2.3 rns\_double() [3/4]**

```
rns_double (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.258.2.4 rns\_double() [4/4]**

```
rns_double (
    const RNSIntegerMod< rns_double > & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.258.3 Member Function Documentation****16.258.3.1 precompute\_cst()**

```
void precompute_cst (
    size_t K = 0 ) [inline]
```

**16.258.3.2 init() [1/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const T * A,
    size_t lda,
    const integer & maxA,
    bool RNS_MAJOR = false ) const [inline]
```

**16.258.3.3 init() [2/3]**

```
void init (
    size_t m,
    size_t n,
```

```
double * Arns,
size_t rda,
const integer * A,
size_t lda,
size_t k,
bool RNS_MAJOR = false ) const [inline]
```

#### 16.258.3.4 init\_transpose()

```
void init_transpose (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.258.3.5 convert() [1/2]

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.258.3.6 convert\_transpose()

```
void convert_transpose (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.258.3.7 reduce()

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.258.3.8 init() [3/3]

```
void init (
```

```

    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const RecInt::ruint< K > * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]

```

#### 16.258.3.9 convert() [2/2]

```

void convert (
    size_t m,
    size_t n,
    integer gamma,
    RecInt::ruint< K > * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    integer p = 0,
    bool RNS_MAJOR = false ) const [inline]

```

### 16.258.4 Field Documentation

#### 16.258.4.1 \_basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

#### 16.258.4.2 \_basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

#### 16.258.4.3 \_negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

#### 16.258.4.4 \_invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.258.4.5 \_field\_rns

```
std::vector<ModField> _field_rns
```

#### 16.258.4.6 \_M

```
integer _M
```

#### 16.258.4.7 \_Mi

```
std::vector<integer> _Mi
```

#### 16.258.4.8 \_MMi

```
std::vector<double> _MMi
```

#### 16.258.4.9 \_crt\_in

```
std::vector<double> _crt_in
```

#### 16.258.4.10 \_crt\_out

```
std::vector<double> _crt_out
```

#### 16.258.4.11 \_size

```
size_t _size
```

#### 16.258.4.12 \_pbits

```
size_t _pbits
```

#### 16.258.4.13 \_ldm

```
size_t _ldm
```

#### 16.258.4.14 \_mi\_sum

```
integer _mi_sum
```

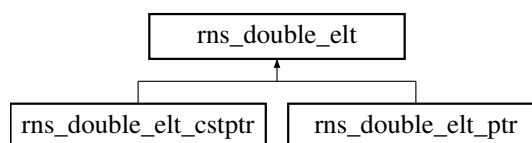
The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double-recint.inl](#)
- [rns-double.inl](#)

## 16.259 rns\_double\_elt Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns\_double\_elt:



## Public Member Functions

- [rns\\_double\\_elt](#) ()
- [~rns\\_double\\_elt](#) ()
- [rns\\_double\\_elt](#) (double \*p, size\_t r, size\_t a=false)
- [rns\\_double\\_elt\\_ptr](#) operator& ()
- [rns\\_double\\_elt\\_cstptr](#) operator& () const
- [rns\\_double\\_elt](#) (const [rns\\_double\\_elt](#) &x)

## Data Fields

- double \* [\\_ptr](#)
- size\_t [\\_stride](#)
- bool [\\_alloc](#)

## 16.259.1 Constructor & Destructor Documentation

### 16.259.1.1 [rns\\_double\\_elt](#)() [1/3]

```
rns\_double\_elt ( ) [inline]
```

### 16.259.1.2 [~rns\\_double\\_elt](#)()

```
~rns\_double\_elt ( ) [inline]
```

### 16.259.1.3 [rns\\_double\\_elt](#)() [2/3]

```
rns\_double\_elt (
    double * p,
    size_t r,
    size_t a = false ) [inline]
```

### 16.259.1.4 [rns\\_double\\_elt](#)() [3/3]

```
rns\_double\_elt (
    const rns\_double\_elt & x ) [inline]
```

## 16.259.2 Member Function Documentation

### 16.259.2.1 [operator&\(\)](#) [1/2]

```
rns\_double\_elt\_ptr operator& ( ) [inline]
```

### 16.259.2.2 [operator&\(\)](#) [2/2]

```
rns\_double\_elt\_cstptr operator& ( ) const [inline]
```

## 16.259.3 Field Documentation

**16.259.3.1 \_ptr**

```
double* _ptr
```

**16.259.3.2 \_stride**

```
size_t _stride
```

**16.259.3.3 \_alloc**

```
bool _alloc
```

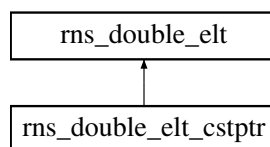
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

**16.260 rns\_double\_elt\_cstptr Struct Reference**

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns\_double\_elt\_cstptr:

**Public Member Functions**

- [rns\\_double\\_elt\\_cstptr](#) ()
- [rns\\_double\\_elt\\_cstptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) ([rns\\_double\\_elt\\_cstptr](#) &&)=default
- [rns\\_double\\_elt\\_cstptr \\* operator&](#) ()
- [rns\\_double\\_elt & operator\\*](#) () const
- [rns\\_double\\_elt operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt & operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_cstptr operator++](#) ()
- [rns\\_double\\_elt\\_cstptr operator--](#) ()
- [rns\\_double\\_elt\\_cstptr operator+](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr operator-](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr & operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr & operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr & operator=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [bool operator<](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [bool operator!=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr operator&](#) () const

**Data Fields**

- [rns\\_double\\_elt](#) other
- [double \\* \\_ptr](#)
- [size\\_t \\_stride](#)
- [bool \\_alloc](#)

## 16.260.1 Constructor & Destructor Documentation

### 16.260.1.1 `rns_double_elt_cstptr()` [1/5]

```
rns_double_elt_cstptr ( ) [inline]
```

### 16.260.1.2 `rns_double_elt_cstptr()` [2/5]

```
rns_double_elt_cstptr (
    double * p,
    size_t r ) [inline]
```

### 16.260.1.3 `rns_double_elt_cstptr()` [3/5]

```
rns_double_elt_cstptr (
    const rns_double_elt_ptr & x ) [inline]
```

### 16.260.1.4 `rns_double_elt_cstptr()` [4/5]

```
rns_double_elt_cstptr (
    const rns_double_elt_cstptr & x ) [inline]
```

### 16.260.1.5 `rns_double_elt_cstptr()` [5/5]

```
rns_double_elt_cstptr (
    rns_double_elt_cstptr && ) [default]
```

## 16.260.2 Member Function Documentation

### 16.260.2.1 `operator&()` [1/2]

```
rns_double_elt_cstptr* operator& ( ) [inline]
```

### 16.260.2.2 `operator*()`

```
rns_double_elt& operator* ( ) const [inline]
```

### 16.260.2.3 `operator[]()` [1/2]

```
rns_double_elt operator[] (
    size_t i ) const [inline]
```

### 16.260.2.4 `operator[]()` [2/2]

```
rns_double_elt& operator[] (
    size_t i ) [inline]
```

**16.260.2.5 operator++()**

```
rns_double_elt_cstptr operator++ ( ) [inline]
```

**16.260.2.6 operator--()**

```
rns_double_elt_cstptr operator-- ( ) [inline]
```

**16.260.2.7 operator+()**

```
rns_double_elt_cstptr operator+ (
    size_t inc ) const [inline]
```

**16.260.2.8 operator-()**

```
rns_double_elt_cstptr operator- (
    size_t inc ) const [inline]
```

**16.260.2.9 operator+=()**

```
rns_double_elt_cstptr& operator+= (
    size_t inc ) [inline]
```

**16.260.2.10 operator-=()**

```
rns_double_elt_cstptr& operator-= (
    size_t inc ) [inline]
```

**16.260.2.11 operator=()**

```
rns_double_elt_cstptr & operator= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.260.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.260.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.260.2.14 operator&() [2/2]**

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

**16.260.3 Field Documentation**

**16.260.3.1 other**

```
rns_double_elt other
```

**16.260.3.2 \_ptr**

```
double* _ptr [inherited]
```

**16.260.3.3 \_stride**

```
size_t _stride [inherited]
```

**16.260.3.4 \_alloc**

```
bool _alloc [inherited]
```

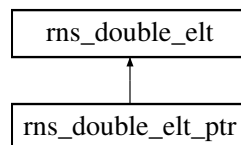
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

**16.261 rns\_double\_elt\_ptr Struct Reference**

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt_ptr`:

**Public Member Functions**

- [rns\\_double\\_elt\\_ptr](#) ()
- [rns\\_double\\_elt\\_ptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) ([rns\\_double\\_elt\\_ptr](#) &&)=default
- [rns\\_double\\_elt\\_ptr](#) \* [operator&](#) ()
- [rns\\_double\\_elt](#) & [operator\\*](#) ()
- [rns\\_double\\_elt](#) [operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt](#) & [operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_ptr](#) [operator++](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator--](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator+](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) [operator-](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- bool [operator<](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- bool [operator!=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) [operator&](#) () const

## Data Fields

- [rns\\_double\\_elt](#) other
- double \* [\\_ptr](#)
- size\_t [\\_stride](#)
- bool [\\_alloc](#)

## 16.261.1 Constructor & Destructor Documentation

### 16.261.1.1 rns\_double\_elt\_ptr() [1/5]

```
rns_double_elt_ptr ( ) [inline]
```

### 16.261.1.2 rns\_double\_elt\_ptr() [2/5]

```
rns_double_elt_ptr (
    double * p,
    size_t r ) [inline]
```

### 16.261.1.3 rns\_double\_elt\_ptr() [3/5]

```
rns_double_elt_ptr (
    const rns_double_elt_ptr & x ) [inline]
```

### 16.261.1.4 rns\_double\_elt\_ptr() [4/5]

```
rns_double_elt_ptr (
    const rns_double_elt_cstptr & x ) [inline]
```

### 16.261.1.5 rns\_double\_elt\_ptr() [5/5]

```
rns_double_elt_ptr (
    rns_double_elt_ptr && ) [default]
```

## 16.261.2 Member Function Documentation

### 16.261.2.1 operator&() [1/2]

```
rns_double_elt_ptr* operator& ( ) [inline]
```

### 16.261.2.2 operator\*()

```
rns_double_elt& operator* ( ) [inline]
```

### 16.261.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
    size_t i ) const [inline]
```

**16.261.2.4 operator[]()** [2/2]

```
rns_double_elt& operator[] (
    size_t i ) [inline]
```

**16.261.2.5 operator++()**

```
rns_double_elt_ptr operator++ ( ) [inline]
```

**16.261.2.6 operator--()**

```
rns_double_elt_ptr operator-- ( ) [inline]
```

**16.261.2.7 operator+()**

```
rns_double_elt_ptr operator+ (
    size_t inc ) [inline]
```

**16.261.2.8 operator-()**

```
rns_double_elt_ptr operator- (
    size_t inc ) [inline]
```

**16.261.2.9 operator+=()**

```
rns_double_elt_ptr& operator+= (
    size_t inc ) [inline]
```

**16.261.2.10 operator-=()**

```
rns_double_elt_ptr& operator-= (
    size_t inc ) [inline]
```

**16.261.2.11 operator=()**

```
rns_double_elt_ptr & operator= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.261.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_ptr & x ) [inline]
```

**16.261.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.261.2.14 operator&()** [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

### 16.261.3 Field Documentation

#### 16.261.3.1 other

`rns_double_elt` other

#### 16.261.3.2 \_ptr

`double* _ptr` [inherited]

#### 16.261.3.3 \_stride

`size_t _stride` [inherited]

#### 16.261.3.4 \_alloc

`bool _alloc` [inherited]

The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.262 rns\_double\_extended Struct Reference

```
#include <rns-double.h>
```

### Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::ModularExtended< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [rns\\_double\\_extended](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double\\_extended](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double\\_extended](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) ()
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false)
- void [convert](#) (size\_t m, size\_t n, [integer](#) gamma, [integer](#) \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false)
- void [init](#) (size\_t m, double \*Arns, const [integer](#) \*A, size\_t lda) const
- void [convert](#) (size\_t m, [integer](#) \*A, const double \*Arns) const
- void [reduce](#) (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const

## Data Fields

- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis`
- `std::vector< ModField > _field_rns`
- `integer _M`
- `std::vector< integer > _Mi`
- `std::vector< double > _MMi`
- `std::vector< double > _crt_in`
- `std::vector< double > _crt_out`
- `size_t _size`
- `size_t _pbits`
- `size_t _ldm`

## 16.262.1 Member Typedef Documentation

### 16.262.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.262.1.2 ModField

```
typedef Givaro::ModularExtended<double> ModField
```

### 16.262.1.3 BasisElement

```
typedef double BasisElement
```

### 16.262.1.4 Element

```
typedef rns\_double\_elt Element
```

### 16.262.1.5 Element\_ptr

```
typedef rns\_double\_elt\_ptr Element\_ptr
```

### 16.262.1.6 ConstElement\_ptr

```
typedef rns\_double\_elt\_cstptr ConstElement\_ptr
```

## 16.262.2 Constructor & Destructor Documentation

**16.262.2.1 rns\_double\_extended() [1/3]**

```
rns_double_extended (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.262.2.2 rns\_double\_extended() [2/3]**

```
rns_double_extended (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

**16.262.2.3 rns\_double\_extended() [3/3]**

```
rns_double_extended (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.262.3 Member Function Documentation****16.262.3.1 precompute\_cst()**

```
void precompute_cst ( ) [inline]
```

**16.262.3.2 init() [1/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    const integer & maxA,
    bool RNS_MAJOR = false ) const [inline]
```

**16.262.3.3 init() [2/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) [inline]
```

**16.262.3.4 convert() [1/2]**

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) [inline]
```

**16.262.3.5 init() [3/3]**

```
void init (
    size_t m,
    double * Arns,
    const integer * A,
    size_t lda ) const [inline]
```

**16.262.3.6 convert() [2/2]**

```
void convert (
    size_t m,
    integer * A,
    const double * Arns ) const [inline]
```

**16.262.3.7 reduce()**

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

**16.262.4 Field Documentation****16.262.4.1 \_basis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

**16.262.4.2 \_basisMax**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

**16.262.4.3 \_negbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

#### 16.262.4.4 `_invbasis`

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.262.4.5 `_field_rns`

```
std::vector<ModField> _field_rns
```

#### 16.262.4.6 `_M`

```
integer _M
```

#### 16.262.4.7 `_Mi`

```
std::vector<integer> _Mi
```

#### 16.262.4.8 `_MMi`

```
std::vector<double> _MMi
```

#### 16.262.4.9 `_crt_in`

```
std::vector<double> _crt_in
```

#### 16.262.4.10 `_crt_out`

```
std::vector<double> _crt_out
```

#### 16.262.4.11 `_size`

```
size_t _size
```

#### 16.262.4.12 `_pbits`

```
size_t _pbits
```

#### 16.262.4.13 `_ldm`

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double.inl](#)

## 16.263 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```

### 16.263.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.264 RNSInteger< RNS > Class Template Reference

```
#include <rns-integer.h>
```

### Data Structures

- class [RandIter](#)

### Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [RNSInteger](#) (const [RNS](#) &myrns)
- template<typename T >  
  [RNSInteger](#) (const T &F)
- const [RNS](#) & [rns](#) () const
- size\_t [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- integer [characteristic](#) ([integer](#) &p) const
- integer [cardinality](#) ([integer](#) &p) const
- [Element](#) & [init](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) & [reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [reduce](#) ([Element](#) &x) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) & [assign](#) ([Element](#) &x, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os) const

### Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

### Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Integer [integer](#)

### Protected Attributes

- const [RNS](#) \* [\\_rns](#)

## 16.264.1 Member Typedef Documentation

### 16.264.1.1 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.264.1.2 integer

```
typedef Givaro::Integer integer [protected]
```

### 16.264.1.3 Element

```
typedef RNS::Element Element
```

### 16.264.1.4 Element\_ptr

```
typedef RNS::Element_ptr Element_ptr
```

### 16.264.1.5 ConstElement\_ptr

```
typedef RNS::ConstElement_ptr ConstElement_ptr
```

## 16.264.2 Constructor & Destructor Documentation

### 16.264.2.1 RNSInteger() [1/2]

```
RNSInteger (
    const RNS & myrns ) [inline]
```

### 16.264.2.2 RNSInteger() [2/2]

```
RNSInteger (
    const T & F ) [inline]
```

## 16.264.3 Member Function Documentation

### 16.264.3.1 rns()

```
const RNS& rns ( ) const [inline]
```

### 16.264.3.2 size()

```
size_t size ( ) const [inline]
```

**16.264.3.3 isOne()**

```
bool isOne (
    const Element & x ) const [inline]
```

**16.264.3.4 isMOne()**

```
bool isMOne (
    const Element & x ) const [inline]
```

**16.264.3.5 isZero()**

```
bool isZero (
    const Element & x ) const [inline]
```

**16.264.3.6 characteristic()**

```
integer characteristic (
    integer & p ) const [inline]
```

**16.264.3.7 cardinality()**

```
integer cardinality (
    integer & p ) const [inline]
```

**16.264.3.8 init() [1/2]**

```
Element& init (
    Element & x ) const [inline]
```

**16.264.3.9 init() [2/2]**

```
Element& init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

**16.264.3.10 reduce() [1/2]**

```
Element& reduce (
    Element & x,
    const Element & y ) const [inline]
```

**16.264.3.11 reduce() [2/2]**

```
Element& reduce (
    Element & x ) const [inline]
```

**16.264.3.12 convert()**

```
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

**16.264.3.13 assign()**

```
Element& assign (
    Element & x,
    const Element & y ) const [inline]
```

**16.264.3.14 write() [1/2]**

```
std::ostream& write (
    std::ostream & os,
    const Element & y ) const [inline]
```

**16.264.3.15 write() [2/2]**

```
std::ostream& write (
    std::ostream & os ) const [inline]
```

**16.264.4 Field Documentation****16.264.4.1 \_rns**

```
const RNS* _rns [protected]
```

**16.264.4.2 one**

```
Element one
```

**16.264.4.3 mOne**

```
Element mOne
```

**16.264.4.4 zero**

```
Element zero
```

The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer.h](#)

**16.265 RNSIntegerMod< RNS > Class Template Reference**

```
#include <rns-integer-mod.h>
```

**Data Structures**

- class [RandIter](#)

## Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

## Public Member Functions

- [RNSIntegerMod](#) (const [integer](#) &p, const [RNS](#) &myrns)
- const [rns\\_double](#) &[rns](#) () const
- const [RNSInteger](#)< [RNS](#) > &[delayed](#) () const
- size\_t [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- [integer](#) &[characteristic](#) ([integer](#) &p) const
- [integer](#) [characteristic](#) () const
- [integer](#) &[cardinality](#) ([integer](#) &p) const
- [integer](#) [cardinality](#) () const
- [integer](#) [minElement](#) () const
- [integer](#) [maxElement](#) () const
- [Element](#) &[init](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) &[reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[reduce](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const [Element](#) &y) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) &[assign](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[add](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[sub](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[neg](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[mul](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[axpyin](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[inv](#) ([Element](#) &x, const [Element](#) &y) const
- bool [areEqual](#) (const [Element](#) &x, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os) const
- void [reduce\\_modp](#) (size\_t n, [Element\\_ptr](#) B) const
- std::ostream &[write\\_matrix](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- std::ostream &[write\\_matrix\\_long](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- void [reduce\\_modp](#) (size\_t m, size\_t n, [Element\\_ptr](#) B, size\_t lda) const
- void [reduce\\_modp\\_rnsmajor](#) (size\_t n, [Element\\_ptr](#) B) const

## Data Fields

- [Element one](#)
- [Element mOne](#)
- [Element zero](#)

## Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Modular< [BasisElement](#) > [ModField](#)
- typedef Givaro::Integer [integer](#)

## Protected Attributes

- [integer \\_p](#)
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _Mi_modp_rns`
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _iM_modp_rns`
- `const RNS * _rns`
- `Givaro::Modular< Givaro::Integer > _F`
- [RNSInteger](#)< [RNS](#) > \_RNSdelayed

## 16.265.1 Member Typedef Documentation

### 16.265.1.1 Element

```
typedef RNS::Element Element
```

### 16.265.1.2 Element\_ptr

```
typedef RNS::Element\_ptr Element_ptr
```

### 16.265.1.3 ConstElement\_ptr

```
typedef RNS::ConstElement\_ptr ConstElement_ptr
```

### 16.265.1.4 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.265.1.5 ModField

```
typedef Givaro::Modular<BasisElement> ModField [protected]
```

### 16.265.1.6 integer

```
typedef Givaro::Integer integer [protected]
```

## 16.265.2 Constructor & Destructor Documentation

### 16.265.2.1 RNSIntegerMod()

```
RNSIntegerMod (
    const integer & p,
    const RNS & myrns ) [inline]
```

## 16.265.3 Member Function Documentation

### 16.265.3.1 rns()

```
const rns\_double& rns ( ) const [inline]
```

**16.265.3.2 delayed()**

```
const RNSInteger<RNS>& delayed ( ) const [inline]
```

**16.265.3.3 size()**

```
size_t size ( ) const [inline]
```

**16.265.3.4 isOne()**

```
bool isOne (
    const Element & x ) const [inline]
```

**16.265.3.5 isMOne()**

```
bool isMOne (
    const Element & x ) const [inline]
```

**16.265.3.6 isZero()**

```
bool isZero (
    const Element & x ) const [inline]
```

**16.265.3.7 characteristic() [1/2]**

```
integer& characteristic (
    integer & p ) const [inline]
```

**16.265.3.8 characteristic() [2/2]**

```
integer characteristic ( ) const [inline]
```

**16.265.3.9 cardinality() [1/2]**

```
integer& cardinality (
    integer & p ) const [inline]
```

**16.265.3.10 cardinality() [2/2]**

```
integer cardinality ( ) const [inline]
```

**16.265.3.11 minElement()**

```
integer minElement ( ) const [inline]
```

**16.265.3.12 maxElement()**

```
integer maxElement ( ) const [inline]
```

**16.265.3.13 init() [1/3]**

```
Element& init (
    Element & x ) const [inline]
```

**16.265.3.14 init() [2/3]**

```
Element& init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

**16.265.3.15 reduce() [1/2]**

```
Element& reduce (
    Element & x,
    const Element & y ) const [inline]
```

**16.265.3.16 reduce() [2/2]**

```
Element& reduce (
    Element & x ) const [inline]
```

**16.265.3.17 init() [3/3]**

```
Element& init (
    Element & x,
    const Element & y ) const [inline]
```

**16.265.3.18 convert()**

```
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

**16.265.3.19 assign()**

```
Element& assign (
    Element & x,
    const Element & y ) const [inline]
```

**16.265.3.20 add()**

```
Element& add (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

**16.265.3.21 sub()**

```
Element& sub (
    Element & x,
```

```
const Element & y,  
const Element & z ) const [inline]
```

#### 16.265.3.22 neg()

```
Element& neg (  
    Element & x,  
    const Element & y ) const [inline]
```

#### 16.265.3.23 mul()

```
Element& mul (  
    Element & x,  
    const Element & y,  
    const Element & z ) const [inline]
```

#### 16.265.3.24 axpyin()

```
Element& axpyin (  
    Element & x,  
    const Element & y,  
    const Element & z ) const [inline]
```

#### 16.265.3.25 inv()

```
Element& inv (  
    Element & x,  
    const Element & y ) const [inline]
```

#### 16.265.3.26 areEqual()

```
bool areEqual (  
    const Element & x,  
    const Element & y ) const [inline]
```

#### 16.265.3.27 write() [1/2]

```
std::ostream& write (  
    std::ostream & os,  
    const Element & y ) const [inline]
```

#### 16.265.3.28 write() [2/2]

```
std::ostream& write (  
    std::ostream & os ) const [inline]
```

#### 16.265.3.29 reduce\_modp() [1/2]

```
void reduce_modp (  
    size_t n,  
    Element_ptr B ) const [inline]
```

**16.265.3.30 write\_matrix()**

```
std::ostream& write_matrix (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.265.3.31 write\_matrix\_long()**

```
std::ostream& write_matrix_long (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.265.3.32 reduce\_modp() [2/2]**

```
void reduce_modp (
    size_t m,
    size_t n,
    Element_ptr B,
    size_t lda ) const [inline]
```

**16.265.3.33 reduce\_modp\_rnsmajor()**

```
void reduce_modp_rnsmajor (
    size_t n,
    Element_ptr B ) const [inline]
```

**16.265.4 Field Documentation****16.265.4.1 \_p**

```
integer _p [protected]
```

**16.265.4.2 \_Mi\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↵
rns [protected]
```

**16.265.4.3 \_iM\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↵
rns [protected]
```

**16.265.4.4 \_rns**

```
const RNS* _rns [protected]
```

**16.265.4.5 \_F**

```
Givaro::Modular<Givaro::Integer> _F [protected]
```

**16.265.4.6 \_RNSdelayed**

```
RNSInteger<RNS> _RNSdelayed [protected]
```

**16.265.4.7 one**

```
Element one
```

**16.265.4.8 mOne**

```
Element mOne
```

**16.265.4.9 zero**

```
Element zero
```

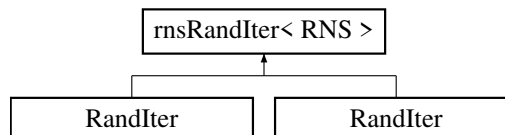
The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer-mod.h](#)

**16.266 rnsRandIter< RNS > Class Template Reference**

```
#include <rns-double.h>
```

Inheritance diagram for rnsRandIter< RNS >:

**Public Member Functions**

- [rnsRandIter](#) (const [RNS](#) &R, uint64\_t seed=0)
- [RNS::Element](#) & [random](#) (typename [RNS::Element](#) &elt) const  
*RNS ring Element random assignement.*
- [RNS::Element](#) & [operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [operator\(\)](#) () const
- [RNS::Element](#) [random](#) () const
- const [RNS](#) & [ring](#) () const

**16.266.1 Constructor & Destructor Documentation****16.266.1.1 rnsRandIter()**

```
rnsRandIter (
    const RNS & R,
    uint64_t seed = 0 ) [inline]
```

## 16.266.2 Member Function Documentation

### 16.266.2.1 random() [1/2]

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline]
```

RNS ring Element random assignment.

Element is supposed to be initialized

Returns

random ring Element

### 16.266.2.2 operator>() [1/2]

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline]
```

### 16.266.2.3 operator>() [2/2]

```
RNS::Element operator() ( ) const [inline]
```

### 16.266.2.4 random() [2/2]

```
RNS::Element random ( ) const [inline]
```

### 16.266.2.5 ring()

```
const RNS& ring ( ) const [inline]
```

The documentation for this class was generated from the following file:

- [rns-double.h](#)

## 16.267 Row Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

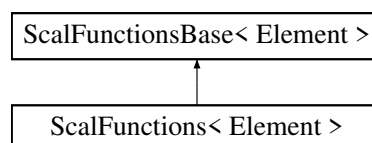
## 16.268 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.269 ScalFunctions< Element > Struct Template Reference

Inheritance diagram for ScalFunctions< Element >:



## Public Types

- using `vectElt` = `vector< Element >`

## Static Public Member Functions

- static void `genInputs` (`vector< vectElt > &inputs`)
- static void `genInputsWithZero` (`vector< vectElt > &inputs`)
- static `Element zero` ()
- static `Element vand` (`Element x1, Element x2`)
- static `Element vor` (`Element x1, Element x2`)
- static `Element vxor` (`Element x1, Element x2`)
- static `Element vandnot` (`Element x1, Element x2`)
- static `Element add` (`Element x1, Element x2`)
- static `Element addin` (`Element &x1, Element x2`)
- static `Element sub` (`Element x1, Element x2`)
- static `Element subin` (`Element &x1, Element x2`)
- static `Element mul` (`Element x1, Element x2`)
- static `Element mulin` (`Element &x1, Element x2`)
- static `Element div` (`Element x1, Element x2`)
- static `Element fmadd` (`Element x1, Element x2, Element x3`)
- static `Element fmaddin` (`Element &x1, Element x2, Element x3`)
- static `Element fmsub` (`Element x1, Element x2, Element x3`)
- static `Element fmsubin` (`Element &x1, Element x2, Element x3`)
- static `Element fnmadd` (`Element x1, Element x2, Element x3`)
- static `Element fnmaddin` (`Element &x1, Element x2, Element x3`)
- static `Element lesser` (`Element x1, Element x2`)
- static `Element lesser_eq` (`Element x1, Element x2`)
- static `Element greater` (`Element x1, Element x2`)
- static `Element greater_eq` (`Element x1, Element x2`)
- static `Element eq` (`Element x1, Element x2`)
- static `vectElt unpacklo` (`vectElt a, vectElt b`)
- static `vectElt unpackhi` (`vectElt a, vectElt b`)
- static void `unpacklohi` (`vectElt &lo, vectElt &hi, vectElt a, vectElt b`)
- static `vectElt pack_even` (`vectElt a, vectElt b`)
- static `vectElt pack_odd` (`vectElt a, vectElt b`)
- static void `pack` (`vectElt &even, vectElt &odd, vectElt a, vectElt b`)
- template<uint16\_t s>  
static `vectElt blend` (`vectElt a, vectElt b`)

## 16.269.1 Member Typedef Documentation

### 16.269.1.1 vectElt

using `vectElt` = `vector<Element>`

## 16.269.2 Member Function Documentation

### 16.269.2.1 genInputs()

```
static void genInputs (
    vector< vectElt > & inputs ) [inline], [static]
```

**16.269.2.2 genInputsWithZero()**

```
static void genInputsWithZero (
    vector< vectElt > & inputs ) [inline], [static]
```

**16.269.2.3 zero()**

```
static Element zero ( ) [inline], [static]
```

**16.269.2.4 vand()**

```
static Element vand (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.5 vor()**

```
static Element vor (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.6 vxor()**

```
static Element vxor (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.7 vandnot()**

```
static Element vandnot (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.8 add()**

```
static Element add (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.9 addin()**

```
static Element addin (
    Element & x1,
    Element x2 ) [inline], [static]
```

**16.269.2.10 sub()**

```
static Element sub (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.269.2.11 subin()**

```
static Element subin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.12 mul()**

```
static Element mul (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.13 mulin()**

```
static Element mulin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.14 div()**

```
static Element div (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.15 fmadd()**

```
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.16 fmaddin()**

```
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.17 fmsub()**

```
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.18 fmsubin()**

```
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.19 fnmadd()**

```
static Element fnmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.20 fnmaddin()**

```
static Element fnmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.269.2.21 lesser()**

```
static Element lesser (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.22 lesser\_eq()**

```
static Element lesser_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.23 greater()**

```
static Element greater (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.24 greater\_eq()**

```
static Element greater_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.25 eq()**

```
static Element eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.269.2.26 unpacklo()**

```
static vectElt unpacklo (  
    vectElt a,  
    vectElt b ) [inline], [static]
```

**16.269.2.27 unpackhi()**

```
static vectElt unpackhi (
    vectElt a,
    vectElt b ) [inline], [static]
```

**16.269.2.28 unpacklohi()**

```
static void unpacklohi (
    vectElt & lo,
    vectElt & hi,
    vectElt a,
    vectElt b ) [inline], [static]
```

**16.269.2.29 pack\_even()**

```
static vectElt pack_even (
    vectElt a,
    vectElt b ) [inline], [static]
```

**16.269.2.30 pack\_odd()**

```
static vectElt pack_odd (
    vectElt a,
    vectElt b ) [inline], [static]
```

**16.269.2.31 pack()**

```
static void pack (
    vectElt & even,
    vectElt & odd,
    vectElt a,
    vectElt b ) [inline], [static]
```

**16.269.2.32 blend()**

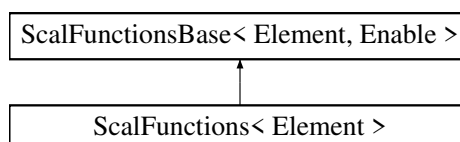
```
static vectElt blend (
    vectElt a,
    vectElt b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.270 ScalFunctionsBase< Element, Enable > Struct Template Reference

Inheritance diagram for ScalFunctionsBase< Element, Enable >:



The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.271 ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type > Struct Template Reference

### Data Structures

- class [FloatingPointTestDistribution](#)

### Static Public Member Functions

- static FloatingPointTestDistribution [get\\_default\\_random\\_generator](#) ()
- static Element [ceil](#) (Element x)
- static Element [floor](#) (Element x)
- static Element [round](#) (Element x)
- static Element [blendv](#) (Element a, Element b, Element mask)
- static Element [fma](#) (Element x, Element y, Element z)

### Static Public Attributes

- static constexpr Element [\\_zero](#) = 0.0
- static constexpr Element [cmp\\_true](#) = NAN
- static constexpr Element [cmp\\_false](#) = [\\_zero](#)

## 16.271.1 Member Function Documentation

### 16.271.1.1 [get\\_default\\_random\\_generator\(\)](#)

```
static FloatingPointTestDistribution get_default_random_generator ( ) [inline], [static]
```

### 16.271.1.2 [ceil\(\)](#)

```
static Element ceil (
    Element x ) [inline], [static]
```

### 16.271.1.3 [floor\(\)](#)

```
static Element floor (
    Element x ) [inline], [static]
```

### 16.271.1.4 [round\(\)](#)

```
static Element round (
    Element x ) [inline], [static]
```

**16.271.1.5 blendv()**

```
static Element blendv (
    Element a,
    Element b,
    Element mask ) [inline], [static]
```

**16.271.1.6 fma()**

```
static Element fma (
    Element x,
    Element y,
    Element z ) [inline], [static]
```

**16.271.2 Field Documentation****16.271.2.1 \_zero**

```
constexpr Element _zero = 0.0 [static], [constexpr]
```

**16.271.2.2 cmp\_true**

```
constexpr Element cmp_true = NAN [static], [constexpr]
```

**16.271.2.3 cmp\_false**

```
constexpr Element cmp_false = _zero [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.272 ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type > Struct Template Reference

**Static Public Member Functions**

- static uniform\_int\_distribution< Element > [get\\_default\\_random\\_generator](#) ()
- static Element [round](#) (Element x)
- static Element [fma](#) (Element x, Element y, Element z)
- static Element [mullo](#) (Element x1, Element x2)
- static Element [mulhi](#) (Element x1, Element x2)
- static Element [mulx](#) (Element x1, Element x2)
- static Element [fmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fmaddxin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsubx](#) (Element x1, Element x2, Element x3)
- static Element [fmsubxin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddxin](#) (Element &x1, Element x2, Element x3)
- template<int s>  
static Element [sra](#) (Element x1)
- template<int s>  
static Element [srl](#) (Element x1)
- template<int s>  
static Element [sll](#) (Element x1)

## Static Public Attributes

- static constexpr Element [\\_zero](#) = 0
- static constexpr Element [cmp\\_true](#) = -1
- static constexpr Element [cmp\\_false](#) = [\\_zero](#)

## 16.272.1 Member Function Documentation

### 16.272.1.1 [get\\_default\\_random\\_generator\(\)](#)

```
static uniform_int_distribution<Element> get_default_random_generator ( ) [inline], [static]
```

### 16.272.1.2 [round\(\)](#)

```
static Element round (  
    Element x ) [inline], [static]
```

### 16.272.1.3 [fma\(\)](#)

```
static Element fma (  
    Element x,  
    Element y,  
    Element z ) [inline], [static]
```

### 16.272.1.4 [mullo\(\)](#)

```
static Element mullo (  
    Element x1,  
    Element x2 ) [inline], [static]
```

### 16.272.1.5 [mulhi\(\)](#)

```
static Element mulhi (  
    Element x1,  
    Element x2 ) [inline], [static]
```

### 16.272.1.6 [mulx\(\)](#)

```
static Element mulx (  
    Element x1,  
    Element x2 ) [inline], [static]
```

### 16.272.1.7 [fmaddx\(\)](#)

```
static Element fmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.8 fmaddxin()**

```
static Element fmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.9 fmsubx()**

```
static Element fmsubx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.10 fmsubxin()**

```
static Element fmsubxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.11 fnmaddx()**

```
static Element fnmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.12 fnmaddxin()**

```
static Element fnmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.272.1.13 sra()**

```
static Element sra (  
    Element x1 ) [inline], [static]
```

**16.272.1.14 srl()**

```
static Element srl (  
    Element x1 ) [inline], [static]
```

**16.272.1.15 sll()**

```
static Element sll (  
    Element x1 ) [inline], [static]
```

**16.272.2 Field Documentation**

**16.272.2.1 `_zero`**

```
constexpr Element _zero = 0 [static], [constexpr]
```

**16.272.2.2 `cmp_true`**

```
constexpr Element cmp_true = -1 [static], [constexpr]
```

**16.272.2.3 `cmp_false`**

```
constexpr Element cmp_false = \_zero [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

**16.273 Sequential Struct Reference****Public Member Functions**

- [Sequential](#) ()
- [Sequential](#) (size\_t nth)
- template<class Cut, class Param >  
  [Sequential](#) ([Parallel](#)< Cut, Param > &)
- size\_t [numthreads](#) () const

**Friends**

- std::ostream & [operator<<](#) (std::ostream &out, const [Sequential](#) &)

**16.273.1 Constructor & Destructor Documentation****16.273.1.1 `Sequential()` [1/3]**

```
Sequential ( ) [inline]
```

**16.273.1.2 `Sequential()` [2/3]**

```
Sequential (
    size_t nth ) [inline]
```

**16.273.1.3 `Sequential()` [3/3]**

```
Sequential (
    Parallel< Cut, Param > & ) [inline]
```

**16.273.2 Member Function Documentation****16.273.2.1 `numthreads()`**

```
size_t numthreads ( ) const [inline]
```

### 16.273.3 Friends And Related Function Documentation

#### 16.273.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Sequential & ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.274 Simd128\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

### 16.275 Simd128\_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- [simd128\\_float.inl](#)

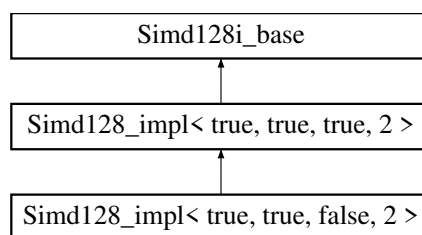
### 16.276 Simd128\_impl< true, false, true, 8 > Struct Reference

The documentation for this struct was generated from the following file:

- [simd128\\_double.inl](#)

### 16.277 Simd128\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, false, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = uint16\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename [Field::Element](#), [scalar\\_t](#) >
- using [vect\\_t](#) = \_\_m128i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmadx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmadxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmadx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmadxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<uint32\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_odd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [pack](#) ([vect\\_t](#) &even, [vect\\_t](#) &odd, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [transpose](#) ([vect\\_t](#) &r0, [vect\\_t](#) &r1, [vect\\_t](#) &r2, [vect\\_t](#) &r3, [vect\\_t](#) &r4, [vect\\_t](#) &r5, [vect\\_t](#) &r6, [vect\\_t](#) &r7)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `__m64` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE CONST vect_t zero` ()
- template<uint8\_t s>  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 16

## 16.277.1 Member Typedef Documentation

### 16.277.1.1 scalar\_t

```
using scalar_t = uint16_t
```

### 16.277.1.2 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.277.1.3 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.277.1.4 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

### 16.277.1.5 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.277.2 Member Function Documentation

### 16.277.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.277.2.2 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.277.2.3 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

### 16.277.2.4 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

### 16.277.2.5 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

### 16.277.2.6 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

### 16.277.2.7 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

### 16.277.2.8 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.277.2.9 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.277.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.277.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.277.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.277.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.277.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.277.2.15 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.277.2.16 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.277.2.17 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.277.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.277.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.277.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.277.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.277.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.277.2.23 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.277.2.24 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.277.2.25 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

#### 16.277.2.26 sll()

```
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.277.2.27 srl()

```
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.277.2.28 shuffle()

```
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.277.2.29 unpacklo\_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.277.2.30 unpackhi\_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.277.2.31 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.277.2.32 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.277.2.33 unpacklohi()

```
static INLINE void unpacklohi (
    vect_t & lo,
    vect_t & hi,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.34 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.35 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.36 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.37 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static], [inherited]
```

**16.277.2.38 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.39 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.40 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.41 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.42 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.43 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.44 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.45 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.46 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.47 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.48 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.49 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.50 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.51 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.52 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.277.2.53 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const __m64 & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.277.2.54 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.277.2.55 sll128()**

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.277.2.56 srl128()**

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.277.2.57 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.58 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.59 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.2.60 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.277.3 Field Documentation****16.277.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.277.3.2 alignment**

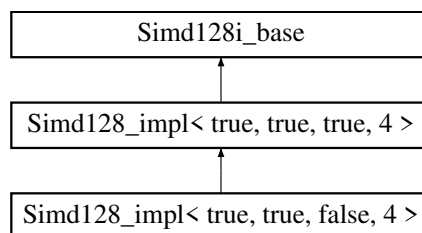
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

**16.278 Simd128\_impl< true, true, false, 4 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, false, 4 >:

**Data Structures**

- union [Converter](#)

## Public Types

- using `scalar_t` = `uint32_t`
- using `aligned_allocator` = `AlignedAllocator< scalar_t, Alignment(alignment)>`
- using `aligned_vector` = `std::vector< scalar_t, aligned_allocator >`
- template<class Field >  
using `is_same_element` = `std::is_same< typename Field::Element, scalar_t >`
- using `vect_t` = `__m128i`

## Static Public Member Functions

- static const std::string `type_string` ()
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &lo, `vect_t` &hi, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_even` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE void pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE void transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3)

- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static constexpr const size_t vect_size = 4`
- `static constexpr const size_t alignment = 16`

## 16.278.1 Member Typedef Documentation

### 16.278.1.1 scalar\_t

using `scalar_t` = `uint32_t`

### 16.278.1.2 aligned\_allocator

using `aligned_allocator` = `AlignedAllocator<scalar_t, Alignment(alignment)>`

### 16.278.1.3 aligned\_vector

using `aligned_vector` = `std::vector<scalar_t, aligned_allocator>`

### 16.278.1.4 is\_same\_element

using `is_same_element` = `std::is_same<typename Field::Element, scalar_t>`

### 16.278.1.5 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.278.2 Member Function Documentation

### 16.278.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.278.2.2 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.278.2.3 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

### 16.278.2.4 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

### 16.278.2.5 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

### 16.278.2.6 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

### 16.278.2.7 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

### 16.278.2.8 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.278.2.9 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.278.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.278.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.278.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.278.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.278.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.278.2.15 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.278.2.16 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.278.2.17 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.278.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.278.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.278.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.278.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.278.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.278.2.23 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.278.2.24 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.278.2.25 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.278.2.26 sll()**

```
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.27 srl()**

```
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.28 shuffle()**

```
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.29 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.30 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.31 unpacklo()**

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.32 unpackhi()**

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.33 unpacklohi()**

```
static INLINE void unpacklohi (
    vect_t & lo,
    vect_t & hi,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.34 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.35 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.36 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.37 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3 ) [inline], [static], [inherited]
```

**16.278.2.38 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.39 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.40 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.41 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.42 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.43 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.44 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.45 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.46 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.47 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.48 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.49 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.50 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.51 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.52 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.53 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.278.2.54 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.278.2.55 sll128()**

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.56 srl128()**

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.278.2.57 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.58 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.59 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.2.60 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.278.3 Field Documentation****16.278.3.1 vect\_size**

```
constexpr const size_t vect_size = 4 [static], [constexpr], [inherited]
```

**16.278.3.2 alignment**

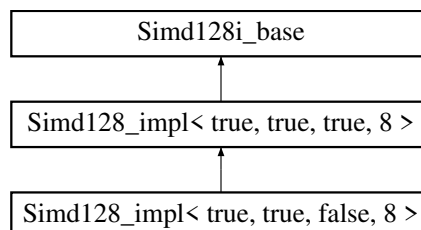
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

**16.279 Simd128\_impl< true, true, false, 8 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, false, 8 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [scalar\\_t](#) = uint64\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >

- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename [Field::Element](#), [scalar\\_t](#) >
- using [vect\\_t](#) = \_\_m128i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) x0, const [vect\\_t](#) x1)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- template<int idx>  
static [INLINE CONST scalar\\_t get](#) ([vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_odd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void pack](#) ([vect\\_t](#) &even, [vect\\_t](#) &odd, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void transpose](#) ([vect\\_t](#) &r0, [vect\\_t](#) &r1)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m128d` &P, const `__m128d` &INVP, const `__m128d` &NEGP, const `vect_t` &POW50REM, const `__m128d` &MIN, const `__m128d` &MAX, `__m128d` &Q, `__m128d` &T)
- static `INLINE CONST vect_t zero` ()
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 2
- static constexpr const size\_t `alignment` = 16

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.279.1 Member Typedef Documentation

### 16.279.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.279.1.2 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.279.1.3 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

#### 16.279.1.4 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

#### 16.279.1.5 vect\_t

```
using vect_t = __m128i [inherited]
```

### 16.279.2 Member Function Documentation

#### 16.279.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

#### 16.279.2.2 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.279.2.3 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1 ) [inline], [static]
```

#### 16.279.2.4 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

#### 16.279.2.5 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

#### 16.279.2.6 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

#### 16.279.2.7 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.279.2.8 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.279.2.9 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.279.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.279.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.279.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.279.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.15 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.279.2.16 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.17 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.18 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.19 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.20 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.21 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.22 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.23 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.279.2.24 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.279.2.25 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.279.2.26 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.279.2.27 get()**

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static], [inherited]
```

**16.279.2.28 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.279.2.29 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.279.2.30 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.279.2.31 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.32 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.33 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.34 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.35 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.36 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.37 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.38 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.39 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1 ) [inline], [static], [inherited]
```

**16.279.2.40 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.41 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.42 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.43 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.44 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.45 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.46 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.47 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.48 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.49 fnmaddin()**

```
static INLINE vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.50 fmsub()**

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.51 fmsubin()**

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.52 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.279.2.53 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.279.2.54 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.279.2.55 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.279.2.56 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INV_P,
    const __m128d & NEG_P,
    const vect_t & POW50REM,
    const __m128d & MIN,
    const __m128d & MAX,
```

```
__m128d & Q,  
__m128d & T ) [static], [inherited]
```

#### 16.279.2.57 signbits()

```
static INLINE CONST vect_t signbits (  
    const vect_t x ) [inline], [static], [protected], [inherited]
```

#### 16.279.2.58 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

#### 16.279.2.59 sll128()

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.279.2.60 srl128()

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.279.2.61 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.279.2.62 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.279.2.63 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.279.2.64 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

### 16.279.3 Field Documentation

**16.279.3.1 vect\_size**

```
constexpr const size_t vect_size = 2 [static], [constexpr], [inherited]
```

**16.279.3.2 alignment**

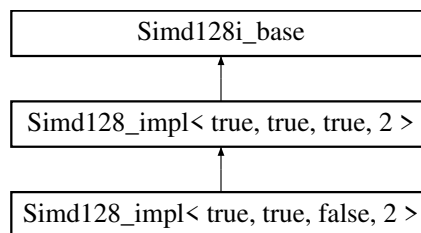
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

**16.280 Simd128\_impl< true, true, true, 2 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, true, 2 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int16\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >

**Static Public Member Functions**

- static const std::string [type\\_string](#) ()
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)

- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t sra (const vect_t a)`
- `template<uint32_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- `static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- `static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 16

## 16.280.1 Member Typedef Documentation

### 16.280.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.280.1.2 scalar\_t

```
using scalar_t = int16_t
```

### 16.280.1.3 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.280.1.4 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.280.1.5 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.280.2 Member Function Documentation

### 16.280.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.280.2.2 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.280.2.3 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.280.2.4 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.280.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

#### 16.280.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.280.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.280.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.280.2.9 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.280.2.10 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.280.2.11 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.280.2.12 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.280.2.13 srl()**

```
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static]
```

**16.280.2.14 sra()**

```
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

**16.280.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static]
```

**16.280.2.16 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.280.2.17 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.280.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.280.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.280.2.20 unpacklohi()**

```
static INLINE void unpacklohi (
    vect_t & lo,
    vect_t & hi,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.280.2.21 pack\_even()**

```
static INLINE CONST vect_t pack_even (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.280.2.22 pack\_odd()

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.23 pack()

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.24 transpose()

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static]
```

#### 16.280.2.25 blend()

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.26 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.27 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.28 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.29 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.30 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.31 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.32 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.33 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.34 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.35 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.36 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.37 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.38 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.39 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.40 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.41 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.42 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.43 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.44 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.280.2.45 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.46 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.47 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.48 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.49 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.50 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.51 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.280.2.52 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.280.2.53 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const __m64 & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.280.2.54 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.280.2.55 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.280.2.56 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.280.2.57 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.280.2.58 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.280.2.59 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.280.2.60 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.280.3 Field Documentation**

### 16.280.3.1 vect\_size

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

### 16.280.3.2 alignment

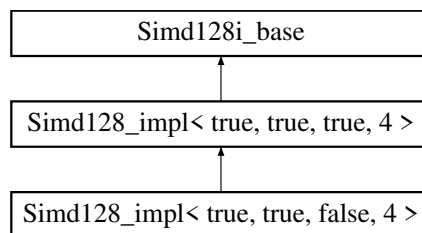
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.281 Simd128\_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int32\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)

- `template<int s>`  
  static `INLINE CONST vect_t srl` (const `vect_t` a)
- `template<int s>`  
  static `INLINE CONST vect_t sra` (const `vect_t` a)
- `template<uint8_t s>`  
  static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &lo, `vect_t` &hi, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_even` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE void pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE void transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3)
- `template<uint8_t s>`  
  static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE CONST vect_t zero` ()
- `template<uint8_t s>`  
  static `INLINE CONST vect_t sll128` (const `vect_t` a)
- `template<uint8_t s>`  
  static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 16

## 16.281.1 Member Typedef Documentation

### 16.281.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.281.1.2 scalar\_t

```
using scalar_t = int32_t
```

### 16.281.1.3 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.281.1.4 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.281.1.5 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.281.2 Member Function Documentation

### 16.281.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.281.2.2 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.281.2.3 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.281.2.4 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.281.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

#### 16.281.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.281.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.281.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.281.2.9 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.281.2.10 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.281.2.11 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.281.2.12 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

#### 16.281.2.13 srl()

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.281.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.281.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.281.2.16 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.17 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.21 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.22 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.23 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.24 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3 ) [inline], [static]
```

**16.281.2.25 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.26 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.27 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.28 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.29 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.30 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.31 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.32 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.33 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.34 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.35 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.36 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.37 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.38 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.39 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.40 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.41 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.42 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.43 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.44 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.45 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.281.2.46 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.281.2.47 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.281.2.48 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.281.2.49 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.281.2.50 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.281.2.51 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.281.2.52 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.281.2.53 mod()

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,
```

```

    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]

```

#### 16.281.2.54 zero()

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

#### 16.281.2.55 sll128()

```

static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]

```

#### 16.281.2.56 srl128()

```

static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]

```

#### 16.281.2.57 vand()

```

static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

#### 16.281.2.58 vor()

```

static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

#### 16.281.2.59 vxor()

```

static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

#### 16.281.2.60 vandnot()

```

static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

### 16.281.3 Field Documentation

#### 16.281.3.1 vect\_size

```

constexpr const size_t vect_size = 4 [static], [constexpr]

```

### 16.281.3.2 alignment

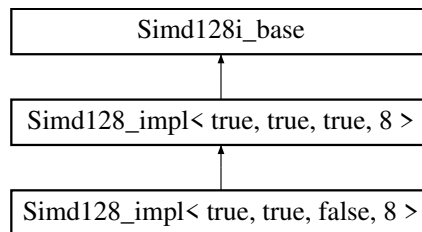
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.282 Simd128\_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- template<int idx>  
static [INLINE CONST scalar\\_t get](#) (vect\_t v)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, vect\_t v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, vect\_t v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const vect\_t v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const vect\_t a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const vect\_t a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const vect\_t a)

- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- `static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- `static INLINE void transpose (vect_t &r0, vect_t &r1)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmadin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmadxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`
- `static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- `static INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 2
- static constexpr const size\_t `alignment` = 16

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.282.1 Member Typedef Documentation

### 16.282.1.1 `vect_t`

```
using vect_t = __m128i
```

### 16.282.1.2 `scalar_t`

```
using scalar_t = int64_t
```

### 16.282.1.3 `aligned_allocator`

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.282.1.4 `aligned_vector`

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.282.1.5 `is_same_element`

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.282.2 Member Function Documentation

### 16.282.2.1 `type_string()`

```
static const std::string type_string ( ) [inline], [static]
```

### 16.282.2.2 `valid()`

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.282.2.3 `compliant()`

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.282.2.4 set1()**

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

**16.282.2.5 set()**

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

**16.282.2.6 gather()**

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.282.2.7 get()**

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

**16.282.2.8 load()**

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.9 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.10 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.11 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.12 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.282.2.13 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.282.2.14 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.282.2.15 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.282.2.16 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.282.2.17 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.18 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.19 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.20 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.21 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.22 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.23 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.24 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.25 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1 ) [inline], [static]
```

**16.282.2.26 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.27 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.28 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.29 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.30 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.31 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.282.2.32 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.33 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.34 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.35 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.36 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.37 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.38 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.41 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.42 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.43 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.44 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.45 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.282.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.47 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.48 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.49 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.50 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.51 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.52 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.282.2.53 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.282.2.54 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.282.2.55 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

**16.282.2.56 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INVP,
    const __m128d & NEGP,
    const vect_t & POW5OREM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static]
```

**16.282.2.57 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

**16.282.2.58 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.282.2.59 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.282.2.60 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.282.2.61 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.62 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.63 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.64 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.3 Field Documentation****16.282.3.1 vect\_size**

```
constexpr const size_t vect_size = 2 [static], [constexpr]
```

**16.282.3.2 alignment**

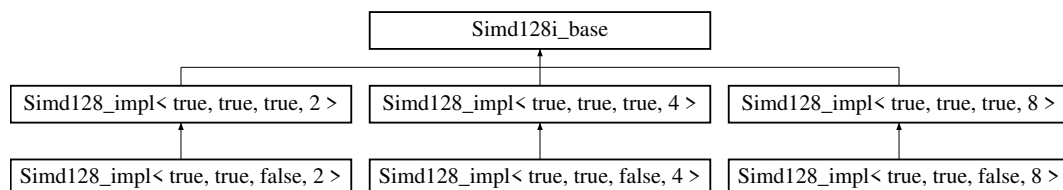
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

**16.283 Simd128i\_base Struct Reference**

Inheritance diagram for Simd128i\_base:

**Public Types**

- using `vect_t` = `__m128i`

**Static Public Member Functions**

- static `INLINE CONST vect_t zero` ()
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128` (const `vect_t` a)

- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## 16.283.1 Member Typedef Documentation

### 16.283.1.1 `vect_t`

using `vect_t` = `__m128i`

## 16.283.2 Member Function Documentation

### 16.283.2.1 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.283.2.2 `sll128()`

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static]
```

### 16.283.2.3 `srl128()`

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static]
```

### 16.283.2.4 `vand()`

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.283.2.5 `vor()`

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.283.2.6 `vxor()`

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.283.2.7 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

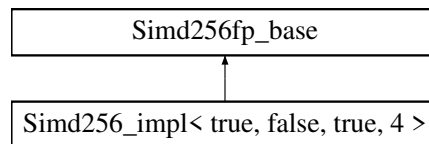
**16.284 Simd256\_impl< ArithType, Int, Signed, Size > Struct Template Reference**

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

**16.285 Simd256\_impl< true, false, true, 4 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, false, true, 4 >:

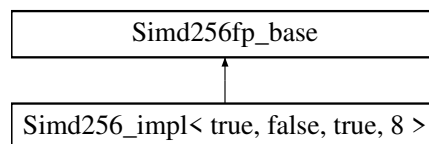


The documentation for this struct was generated from the following file:

- [simd256\\_float.inl](#)

**16.286 Simd256\_impl< true, false, true, 8 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, false, true, 8 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m256d
- using [scalar\\_t](#) = double
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t zero](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE void storeu](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE void stream](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE CONST vect\\_t unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_odd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void pack](#) ([vect\\_t](#) &even, [vect\\_t](#) &odd, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void transpose](#) ([vect\\_t](#) &r0, [vect\\_t](#) &r1, [vect\\_t](#) &r2, [vect\\_t](#) &r3)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t blendv](#) (const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) mask)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t div](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsub](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vand](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vxor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vandnot](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t floor](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t ceil](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t round](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t hadd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- static [INLINE vect\\_t mod](#) ([vect\\_t](#) &C, const [vect\\_t](#) &P, const [vect\\_t](#) &INVP, const [vect\\_t](#) &NEGP, const [vect\\_t](#) &MIN, const [vect\\_t](#) &MAX, [vect\\_t](#) &Q, [vect\\_t](#) &T)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## 16.286.1 Member Typedef Documentation

### 16.286.1.1 `vect_t`

```
using vect_t = __m256d
```

### 16.286.1.2 `scalar_t`

```
using scalar_t = double
```

### 16.286.1.3 `aligned_allocator`

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.286.1.4 `aligned_vector`

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.286.1.5 `is_same_element`

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.286.2 Member Function Documentation

### 16.286.2.1 `type_string()`

```
static const std::string type_string ( ) [inline], [static]
```

### 16.286.2.2 `valid()`

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.286.2.3 `compliant()`

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.286.2.4 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

#### 16.286.2.5 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

#### 16.286.2.6 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4 ) [inline], [static]
```

#### 16.286.2.7 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.286.2.8 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.286.2.9 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.286.2.10 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.286.2.11 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.286.2.12 stream()

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.286.2.13 unpacklo\_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.14 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.15 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.16 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.17 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.18 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.19 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.20 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.21 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,
```

```
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3 ) [inline], [static]
```

#### 16.286.2.22 blend()

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.23 blendv()

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

#### 16.286.2.24 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.25 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.26 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.27 subin()

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.28 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.29 mulin()

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.30 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.31 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.32 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.33 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.34 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.35 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.36 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.37 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.38 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.39 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.40 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.41 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.42 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.43 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.286.2.44 vxor()**

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.45 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.46 floor()

```
static INLINE CONST vect_t floor (  
    const vect_t a ) [inline], [static]
```

#### 16.286.2.47 ceil()

```
static INLINE CONST vect_t ceil (  
    const vect_t a ) [inline], [static]
```

#### 16.286.2.48 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.286.2.49 hadd()

```
static INLINE CONST vect_t hadd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.286.2.50 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.286.2.51 mod()

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

### 16.286.3 Field Documentation

#### 16.286.3.1 vect\_size

```
constexpr const size_t vect_size = 4 [static], [constexpr]
```

### 16.286.3.2 alignment

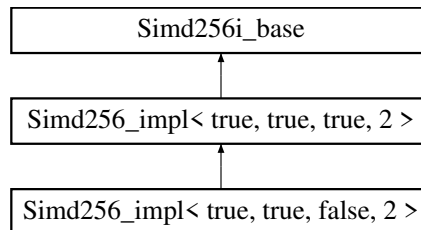
```
constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_double.inl](#)

## 16.287 Simd256\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 2 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = uint16\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >
- using [simdHalf](#) = Simd128< [scalar\\_t](#) >
- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t mulx (vect_t a, vect_t b)`
- static `INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `template<class T >`  
static `constexpr bool valid (T *p)`
- `template<class T >`  
static `constexpr bool compliant (T n)`
- `template<int s>`  
static `INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
static `INLINE CONST vect_t srl (const vect_t a)`
- `template<uint64_t s>`  
static `INLINE CONST vect_t shuffle (const vect_t a)`
- static `INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- static `INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- static `INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- static `INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)`
- `template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s >> 8)>::type * = nullptr>`  
static `INLINE vect_t blend (const vect_t a, const vect_t b)`
- `template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s >> 8)>::type * = nullptr>`  
static `INLINE vect_t blend (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- static `INLINE vect_t addin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- static `INLINE vect_t subin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static `INLINE CONST vect_t zero ()`

## Static Public Attributes

- static `constexpr const size_t vect_size = 16`
- static `constexpr const size_t alignment = 32`

## 16.287.1 Member Typedef Documentation

### 16.287.1.1 scalar\_t

```
using scalar_t = uint16_t
```

### 16.287.1.2 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.287.1.3 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.287.1.4 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

### 16.287.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.287.1.6 vect\_t

```
using vect_t = __m256i [inherited]
```

### 16.287.1.7 half\_t

```
using half_t = __m128i [inherited]
```

## 16.287.2 Member Function Documentation

### 16.287.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.287.2.2 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.287.2.3 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
```

```
const scalar_t x4,  
const scalar_t x5,  
const scalar_t x6,  
const scalar_t x7,  
const scalar_t x8,  
const scalar_t x9,  
const scalar_t x10,  
const scalar_t x11,  
const scalar_t x12,  
const scalar_t x13,  
const scalar_t x14,  
const scalar_t x15 ) [inline], [static]
```

#### 16.287.2.4 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.287.2.5 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.287.2.6 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.287.2.7 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.287.2.8 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.287.2.9 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.287.2.10 sra()

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.287.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.287.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.287.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.15 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.16 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.287.2.17 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.18 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.19 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.20 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.21 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.22 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.287.2.23 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.287.2.24 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.287.2.25 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.287.2.26 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.287.2.27 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.287.2.28 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.287.2.29 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.30 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.31 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.32 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.33 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.34 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.35 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.36 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.37 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7,  
    vect_t & r8,  
    vect_t & r9,  
    vect_t & r10,  
    vect_t & r11,  
    vect_t & r12,  
    vect_t & r13,  
    vect_t & r14,  
    vect_t & r15 ) [inline], [static], [inherited]
```

**16.287.2.38 blend() [1/2]**

```
static INLINE vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.39 blend() [2/2]**

```
static INLINE vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.40 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.41 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.42 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.43 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.44 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.45 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.46 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.47 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.48 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.49 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.50 fmsub()**

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.51 fmsubin()**

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.52 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.287.2.53 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.287.2.54 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

**16.287.2.55 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.287.3 Field Documentation****16.287.3.1 vect\_size**

```
constexpr const size_t vect_size = 16 [static], [constexpr], [inherited]
```

**16.287.3.2 alignment**

```
constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)



- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >
  - static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>
  - static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >
  - static constexpr bool `valid` (T \*p)
- template<class T >
  - static constexpr bool `valid` (T \*p)
- template<class T >
  - static constexpr bool `compliant` (T n)
- template<class T >
  - static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<int s>
  - static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
  - static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
  - static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>
  - static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>
  - static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint8\_t s>
  - static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)

- `template<uint32_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `template<uint64_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- `static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- `static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- `static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)`
- `static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `template<uint16_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t round (const vect_t a)`

- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 32

## 16.288.1 Member Typedef Documentation

### 16.288.1.1 scalar\_t [1/2]

```
using scalar_t = uint32_t
```

### 16.288.1.2 aligned\_allocator [1/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.288.1.3 aligned\_vector [1/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.288.1.4 is\_same\_element [1/2]

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

### 16.288.1.5 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

### 16.288.1.6 scalar\_t [2/2]

```
using scalar_t = uint32_t
```

### 16.288.1.7 aligned\_allocator [2/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.288.1.8 aligned\_vector [2/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.288.1.9 is\_same\_element [2/2]**

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

**16.288.1.10 simdHalf [2/2]**

```
using simdHalf = Simd128<scalar_t>
```

**16.288.1.11 vect\_t [1/2]**

```
using vect_t = __m256i [inherited]
```

**16.288.1.12 vect\_t [2/2]**

```
using vect_t = __m512i [inherited]
```

**16.288.1.13 half\_t [1/2]**

```
using half_t = __m128i [inherited]
```

**16.288.1.14 half\_t [2/2]**

```
using half_t = __m256i [inherited]
```

**16.288.2 Member Function Documentation****16.288.2.1 type\_string() [1/2]**

```
static const std::string type_string ( ) [inline], [static]
```

**16.288.2.2 set1() [1/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.288.2.3 set() [1/3]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.288.2.4 gather()** [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.288.2.5 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.288.2.6 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.288.2.7 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.288.2.8 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.288.2.9 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.288.2.10 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.288.2.11 greater()** [1/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.12 lesser()** [1/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.13 greater\_eq()** [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.14 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.15 mulhi()** [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.16 mulx()** [1/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.17 fmaddx()** [1/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.18 fmaddxin()** [1/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.19 fnmaddx()** [1/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.20 fnmaddxin()** [1/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.21 fmsubx()** [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.22 fmsubxin()** [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.23 hadd\_to\_scal()** [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.288.2.24 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static]
```

**16.288.2.25 set1()** [2/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

**16.288.2.26 set()** [2/3]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

**16.288.2.27 gather()** [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.288.2.28 load()** [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.288.2.29 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.288.2.30 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.288.2.31 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.288.2.32 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.288.2.33 sra()** [2/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.288.2.34 greater()** [2/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.35 lesser()** [2/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.36 greater\_eq()** [2/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.37 lesser\_eq()** [2/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.38 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.39 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.288.2.40 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.41 fmaddxin()** [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.42 fnmaddx()** [2/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.43 fnmaddxin()** [2/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.44 fmsubx()** [2/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.45 fmsubxin()** [2/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.288.2.46 hadd\_to\_scal() [2/2]**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.288.2.47 valid() [1/2]**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.288.2.48 valid() [2/2]**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.288.2.49 compliant() [1/2]**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.288.2.50 compliant() [2/2]**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.288.2.51 set() [3/3]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static], [inherited]
```

**16.288.2.52 sll() [1/2]**

```
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.53 sll()** [2/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.54 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.55 srl()** [2/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.56 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.57 shuffle\_twice()** [2/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.58 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.59 shuffle()** [2/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.60 unpacklo\_intrinsic()** [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.61 unpacklo\_intrinsic()** [2/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.62 unpackhi\_intrinsic() [1/2]**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.63 unpackhi\_intrinsic() [2/2]**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.64 unpacklo() [1/2]**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.65 unpacklo() [2/2]**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.66 unpackhi() [1/2]**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.67 unpackhi() [2/2]**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.68 unpacklohi() [1/2]**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.69 unpacklohi() [2/2]**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.70 pack\_even() [1/2]**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.71 pack\_even() [2/2]**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.72 pack\_odd() [1/2]**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.73 pack\_odd() [2/2]**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.74 pack() [1/2]**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.75 pack() [2/2]**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.76 transpose() [1/2]**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static], [inherited]
```

**16.288.2.77 transpose() [2/2]**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7,  
    vect_t & r8,  
    vect_t & r9,  
    vect_t & r10,  
    vect_t & r11,  
    vect_t & r12,  
    vect_t & r13,  
    vect_t & r14,  
    vect_t & r15 ) [inline], [static], [inherited]
```

**16.288.2.78 blend() [1/2]**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.79 blend() [2/2]**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.80 add() [1/2]**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.81 add() [2/2]**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.82 addin() [1/2]**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.83 addin() [2/2]**

```
static INLINE vect_t addin (  
    vect_t & a,
```

```
const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.84 sub() [1/2]**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.85 sub() [2/2]**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.86 subin() [1/2]**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.87 subin() [2/2]**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.88 mullo() [1/2]**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.89 mullo() [2/2]**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.90 mul() [1/2]**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.91 mul() [2/2]**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.92 fmadd()** [1/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.93 fmadd()** [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.94 fmaddin()** [1/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.95 fmaddin()** [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.96 fnmadd()** [1/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.97 fnmadd()** [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.98 fnmaddin()** [1/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.99 fnmaddin()** [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.100 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.101 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.102 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.103 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.104 eq() [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.105 eq() [2/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.288.2.106 round() [1/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.107 round() [2/2]**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.288.2.108 mod() [1/2]**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.288.2.109 mod() [2/2]**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.288.2.110 zero() [1/2]**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.288.2.111 zero() [2/2]**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.288.2.112 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.113 vxor()**

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.288.2.114 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,
```

```
const vect_t b ) [inline], [static], [inherited]
```

### 16.288.2.115 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

## 16.288.3 Field Documentation

### 16.288.3.1 vect\_size

```
static constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

### 16.288.3.2 alignment

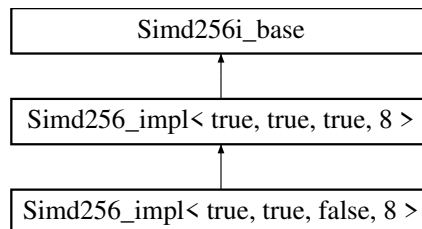
```
static constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.289 Simd256\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 8 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = uint64\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >
  - using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >
- using [simdHalf](#) = Simd128< [scalar\\_t](#) >
- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >
  - static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>
  - static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >
  - static constexpr bool [valid](#) (T \*p)
- template<class T >
  - static constexpr bool [compliant](#) (T n)
- template<int idx>
  - static [INLINE CONST scalar\\_t get](#) ([vect\\_t](#) v)
- template<int s>
  - static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>
  - static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<uint8\_t s>
  - static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_odd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [pack](#) ([vect\\_t](#) &even, [vect\\_t](#) &odd, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [transpose](#) ([vect\\_t](#) &r0, [vect\\_t](#) &r1, [vect\\_t](#) &r2, [vect\\_t](#) &r3)
- template<uint8\_t s>
  - static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m256d` &P, const `__m256d` &INVP, const `__m256d` &NEGP, const `vect_t` &POW50REM, const `__m256d` &MIN, const `__m256d` &MAX, `__m256d` &Q, `__m256d` &T)
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.289.1 Member Typedef Documentation

### 16.289.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.289.1.2 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.289.1.3 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.289.1.4 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

### 16.289.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.289.1.6 vect\_t

```
using vect_t = __m256i [inherited]
```

### 16.289.1.7 half\_t

```
using half_t = __m128i [inherited]
```

## 16.289.2 Member Function Documentation

### 16.289.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.289.2.2 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.289.2.3 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

### 16.289.2.4 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

### 16.289.2.5 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

### 16.289.2.6 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

### 16.289.2.7 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

### 16.289.2.8 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.289.2.9 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.289.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.289.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.289.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.289.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.15 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.289.2.16 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.289.2.17 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.18 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.19 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.20 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.21 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.22 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.23 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.289.2.24 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.289.2.25 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.289.2.26 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.289.2.27 get()**

```
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```

**16.289.2.28 sll()**

```
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

**16.289.2.29 srl()**

```
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

**16.289.2.30 shuffle()**

```
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

**16.289.2.31 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.32 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.33 unpacklo()**

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.34 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.35 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.36 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.37 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.38 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.39 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3 ) [inline], [static], [inherited]
```

**16.289.2.40 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.41 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.42 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.43 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.44 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.45 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.46 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.47 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.48 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.49 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.50 fmsub()**

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.51 fmsubin()**

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.52 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.289.2.53 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.289.2.54 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.289.2.55 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.289.2.56 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INVP,
    const __m256d & NEGP,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static], [inherited]
```

**16.289.2.57 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.289.2.58 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.289.3 Field Documentation****16.289.3.1 vect\_size**

```
constexpr const size_t vect_size = 4 [static], [constexpr], [inherited]
```

**16.289.3.2 alignment**

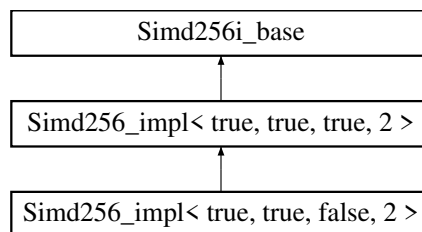
```
constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

**16.290 Simd256\_impl< true, true, true, 2 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, true, true, 2 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int16\_t
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [aligned\\_allocator](#) = [AlignedAllocator](#)< [scalar\\_t](#), [Alignment](#)([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class [Field](#) >
  - using [is\\_same\\_element](#) = std::is\_same< typename [Field::Element](#), [scalar\\_t](#) >

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint64\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t pack\\_odd](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [pack](#) ([vect\\_t](#) &even, [vect\\_t](#) &odd, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [transpose](#) ([vect\\_t](#) &r0, [vect\\_t](#) &r1, [vect\\_t](#) &r2, [vect\\_t](#) &r3, [vect\\_t](#) &r4, [vect\\_t](#) &r5, [vect\\_t](#) &r6, [vect\\_t](#) &r7, [vect\\_t](#) &r8, [vect\\_t](#) &r9, [vect\\_t](#) &r10, [vect\\_t](#) &r11, [vect\\_t](#) &r12, [vect\\_t](#) &r13, [vect\\_t](#) &r14, [vect\\_t](#) &r15)
- template<uint16\_t s, typename std::enable\_if<(s &0x00ff)==(s >> 8)>::type \* = nullptr>  
static [INLINE vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint16\_t s, typename std::enable\_if<(s &0x00ff) !=(s >> 8)>::type \* = nullptr>  
static [INLINE vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) [vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) [vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) [vect\\_t fmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) [vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) [vect\\_t fnmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static `INLINE CONST vect_t zero ()`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 16
- static constexpr const size\_t `alignment` = 32

## 16.290.1 Member Typedef Documentation

### 16.290.1.1 vect\_t

```
using vect_t = __m256i
```

### 16.290.1.2 half\_t

```
using half_t = __m128i
```

### 16.290.1.3 scalar\_t

```
using scalar_t = int16_t
```

### 16.290.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.290.1.5 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.290.1.6 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.290.1.7 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.290.2 Member Function Documentation

### 16.290.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.290.2.2 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.290.2.3 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.290.2.4 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.290.2.5 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

### 16.290.2.6 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

### 16.290.2.7 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.290.2.8 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.290.2.9 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.290.2.10 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.290.2.11 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.290.2.12 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.290.2.13 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.290.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.290.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.290.2.16 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.17 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.21 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.22 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.23 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.24 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,
```

```
vect_t & r4,  
vect_t & r5,  
vect_t & r6,  
vect_t & r7,  
vect_t & r8,  
vect_t & r9,  
vect_t & r10,  
vect_t & r11,  
vect_t & r12,  
vect_t & r13,  
vect_t & r14,  
vect_t & r15 ) [inline], [static]
```

#### 16.290.2.25 blend() [1/2]

```
static INLINE vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.26 blend() [2/2]

```
static INLINE vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.27 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.28 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.29 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.30 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.31 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.32 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.33 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.34 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.290.2.35 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.36 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.37 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.38 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.41 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.42 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.43 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.44 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.45 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.290.2.46 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.290.2.47 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.48 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.49 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.50 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.51 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.290.2.52 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.290.2.53 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.290.2.54 mod()

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,
```

```
const vect_t & MAX,
vect_t & Q,
vect_t & T ) [inline], [static]
```

### 16.290.2.55 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

## 16.290.3 Field Documentation

### 16.290.3.1 vect\_size

```
constexpr const size_t vect_size = 16 [static], [constexpr]
```

### 16.290.3.2 alignment

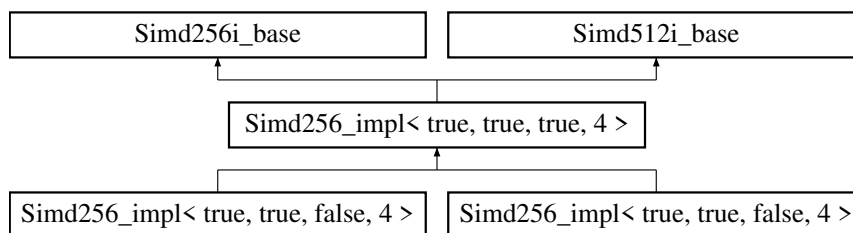
```
constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.291 Simd256\_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, true, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using `vect_t` = `__m256i`
- using `half_t` = `__m128i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `aligned_allocator` = `AlignedAllocator< scalar_t, Alignment(alignment)>`
- using `aligned_vector` = `std::vector< scalar_t, aligned_allocator >`
- template<class `Field` >
  - using `is_same_element` = `std::is_same< typename Field::Element, scalar_t >`
- using `vect_t` = `__m512i`
- using `half_t` = `__m256i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd256< scalar_t >`
- using `aligned_allocator` = `AlignedAllocator< scalar_t, Alignment(alignment)>`

- using `aligned_vector` = `std::vector< scalar_t, aligned_allocator >`
- template<class Field >  
using `is_same_element` = `std::is_same< typename Field::Element, scalar_t >`

## Static Public Member Functions

- static const `std::string type_string` ()
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint32\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &lo, `vect_t` &hi, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_even` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE void pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE void transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3, `vect_t` &r4, `vect_t` &r5, `vect_t` &r6, `vect_t` &r7)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)

- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static const std::string `type_string ()`
- template<class T >  
static constexpr bool `valid (T *p)`
- template<class T >  
static constexpr bool `compliant (T n)`
- static `INLINE CONST vect_t set1 (const scalar_t x)`
- static `INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)`
- template<class T >  
static `INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- static `INLINE PURE vect_t load (const scalar_t *const p)`
- static `INLINE PURE vect_t loadu (const scalar_t *const p)`
- static `INLINE void store (scalar_t *p, vect_t v)`
- static `INLINE void storeu (scalar_t *p, vect_t v)`
- static `INLINE void stream (scalar_t *p, const vect_t v)`
- template<int s>  
static `INLINE CONST vect_t sll (const vect_t a)`
- template<int s>  
static `INLINE CONST vect_t srl (const vect_t a)`
- template<int s>  
static `INLINE CONST vect_t sra (const vect_t a)`
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice (const vect_t a)`
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle (const vect_t a)`
- static `INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- static `INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- static `INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- static `INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)`
- template<uint16\_t s>  
static `INLINE CONST vect_t blend (const vect_t a, const vect_t b)`

- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 32

## 16.291.1 Member Typedef Documentation

### 16.291.1.1 `vect_t` [1/2]

using `vect_t` = `__m256i`

### 16.291.1.2 `half_t` [1/2]

using `half_t` = `__m128i`

**16.291.1.3 scalar\_t [1/2]**

```
using scalar_t = int32_t
```

**16.291.1.4 simdHalf [1/2]**

```
using simdHalf = Simd128<scalar_t>
```

**16.291.1.5 aligned\_allocator [1/2]**

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

**16.291.1.6 aligned\_vector [1/2]**

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.291.1.7 is\_same\_element [1/2]**

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

**16.291.1.8 vect\_t [2/2]**

```
using vect_t = __m512i
```

**16.291.1.9 half\_t [2/2]**

```
using half_t = __m256i
```

**16.291.1.10 scalar\_t [2/2]**

```
using scalar_t = int32_t
```

**16.291.1.11 simdHalf [2/2]**

```
using simdHalf = Simd256<scalar_t>
```

**16.291.1.12 aligned\_allocator [2/2]**

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

**16.291.1.13 aligned\_vector [2/2]**

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.291.1.14 is\_same\_element [2/2]**

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.291.2 Member Function Documentation

### 16.291.2.1 type\_string() [1/2]

```
static const std::string type_string ( ) [inline], [static]
```

### 16.291.2.2 valid() [1/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.291.2.3 compliant() [1/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.291.2.4 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

### 16.291.2.5 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

### 16.291.2.6 gather() [1/2]

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

### 16.291.2.7 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

### 16.291.2.8 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.291.2.9 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.291.2.10 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.291.2.11 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.291.2.12 sll()** [1/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.291.2.13 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.291.2.14 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.291.2.15 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.291.2.16 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.291.2.17 unpacklo\_intrinsic()** [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.18 unpackhi\_intrinsic()** [1/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.19 unpacklo()** [1/2]

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.20 unpackhi()** [1/2]

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.21 unpacklohi()** [1/2]

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.22 pack\_even()** [1/2]

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.23 pack\_odd()** [1/2]

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.24 pack()** [1/2]

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.25 transpose()** [1/2]

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,
```

```
vect_t & r4,  
vect_t & r5,  
vect_t & r6,  
vect_t & r7 ) [inline], [static]
```

#### 16.291.2.26 blend() [1/2]

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.27 add() [1/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.28 addin() [1/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.29 sub() [1/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.30 subin() [1/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.31 mullo() [1/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.32 mul() [1/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.33 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.34 mulx() [1/2]**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.291.2.35 fmadd() [1/2]**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.36 fmaddin() [1/2]**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.37 fmaddx() [1/2]**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.38 fmaddxin() [1/2]**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.39 fnmadd() [1/2]**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.40 fnmaddin() [1/2]**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.41 fnmaddx()** [1/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.42 fnmaddxin()** [1/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.43 fmsub()** [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.44 fmsubin()** [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.45 fmsubx()** [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.46 fmsubxin()** [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.47 eq()** [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.48 greater()** [1/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.49 lesser()** [1/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.50 greater\_eq()** [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.51 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.52 hadd\_to\_scal()** [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.291.2.53 round()** [1/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.291.2.54 mod()** [1/2]

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

**16.291.2.55 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static]
```

**16.291.2.56 valid()** [2/2]

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

**16.291.2.57 compliant()** [2/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.291.2.58 set1()** [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.291.2.59 set()** [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

**16.291.2.60 gather()** [2/2]

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.291.2.61 load()** [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.291.2.62 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.291.2.63 store()** [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.291.2.64 storeu() [2/2]**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.291.2.65 stream() [2/2]**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.291.2.66 sll() [2/2]**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.291.2.67 srl() [2/2]**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.291.2.68 sra() [2/2]**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.291.2.69 shuffle\_twice() [2/2]**

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.291.2.70 shuffle() [2/2]**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.291.2.71 unpacklo\_intrinsic() [2/2]**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.72 unpackhi\_intrinsic() [2/2]**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.73 unpacklo()** [2/2]

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.74 unpackhi()** [2/2]

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.75 unpacklohi()** [2/2]

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.76 pack\_even()** [2/2]

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.77 pack\_odd()** [2/2]

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.78 pack()** [2/2]

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.79 transpose()** [2/2]

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7,  
    vect_t & r8,  
    vect_t & r9,  
    vect_t & r10,
```

```
vect_t & r11,  
vect_t & r12,  
vect_t & r13,  
vect_t & r14,  
vect_t & r15 ) [inline], [static]
```

#### 16.291.2.80 blend() [2/2]

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.81 add() [2/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.82 addin() [2/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.83 sub() [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.84 subin() [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.85 mullo() [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.291.2.86 mul() [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.87 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.88 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.291.2.89 fmadd()** [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.90 fmaddin()** [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.91 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.92 fmaddxin()** [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.93 fnmadd()** [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.94 fnmaddin()** [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.95 fmmaddx()** [2/2]

```
static INLINE CONST vect_t fmmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.96 fmmaddxin()** [2/2]

```
static INLINE vect_t fmmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.97 fmsub()** [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.98 fmsubin()** [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.99 fmsubx()** [2/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.100 fmsubxin()** [2/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.101 eq()** [2/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.102 greater()** [2/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.103 lesser()** [2/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.104 greater\_eq()** [2/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.105 lesser\_eq()** [2/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.291.2.106 hadd\_to\_scal()** [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.291.2.107 round()** [2/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.291.2.108 mod()** [2/2]

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INV_P,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

**16.291.2.109 zero()** [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.291.2.110 zero() [2/2]**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.291.2.111 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.291.2.112 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.291.2.113 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.291.2.114 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.291.3 Field Documentation****16.291.3.1 vect\_size**

```
static constexpr const size_t vect_size = 8 [static], [constexpr]
```

**16.291.3.2 alignment**

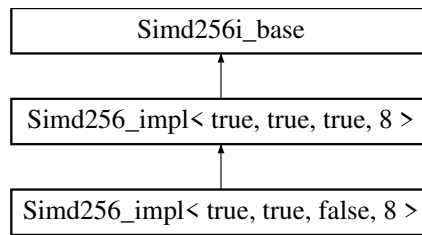
```
static constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

**16.292 Simd256\_impl< true, true, true, 8 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, true, true, 8 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [aligned\\_allocator](#) = [AlignedAllocator](#)< [scalar\\_t](#), [Alignment](#)([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class [Field](#) >  
using [is\\_same\\_element](#) = std::is\_same< typename [Field](#)::[Element](#), [scalar\\_t](#) >

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class [T](#) >  
static constexpr bool [valid](#) ([T](#) \*p)
- template<class [T](#) >  
static constexpr bool [compliant](#) ([T](#) n)
- static [INLINE CONST](#) [vect\\_t](#) [set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST](#) [vect\\_t](#) [set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class [T](#) >  
static [INLINE PURE](#) [vect\\_t](#) [gather](#) (const [scalar\\_t](#) \*const p, const [T](#) \*const idx)
- template<int idx>  
static [INLINE CONST](#) [scalar\\_t](#) [get](#) ([vect\\_t](#) v)
- static [INLINE PURE](#) [vect\\_t](#) [load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE](#) [vect\\_t](#) [loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [sra](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST](#) [vect\\_t](#) [shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST](#) [vect\\_t](#) [unpacklo\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST](#) [vect\\_t](#) [unpackhi\\_intrinsic](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST](#) [vect\\_t](#) [unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST](#) [vect\\_t](#) [unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &lo, [vect\\_t](#) &hi, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST](#) [vect\\_t](#) [pack\\_even](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE` void `pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE` void `transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3)
- template<uint8\_t s>
  - static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const \_\_m256d &P, const \_\_m256d &INVP, const \_\_m256d &NEGP, const `vect_t` &POW50REM, const \_\_m256d &MIN, const \_\_m256d &MAX, \_\_m256d &Q, \_\_m256d &T)
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.292.1 Member Typedef Documentation

### 16.292.1.1 `vect_t`

```
using vect_t = __m256i
```

### 16.292.1.2 half\_t

```
using half_t = __m128i
```

### 16.292.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.292.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.292.1.5 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.292.1.6 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.292.1.7 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.292.2 Member Function Documentation

### 16.292.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.292.2.2 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.292.2.3 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.292.2.4 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.292.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

#### 16.292.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.292.2.7 get()

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

#### 16.292.2.8 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.292.2.9 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.292.2.10 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.292.2.11 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.292.2.12 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.292.2.13 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.292.2.14 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.292.2.15 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.292.2.16 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.292.2.17 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.18 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.19 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.20 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.21 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.22 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.23 pack\_odd()

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.24 pack()

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.25 transpose()

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3 ) [inline], [static]
```

#### 16.292.2.26 blend()

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.27 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.28 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.29 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.292.2.30 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.31 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.292.2.32 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.33 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.292.2.34 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.35 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.36 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.37 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.38 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.41 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.42 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.43 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.44 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.45 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.46 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.47 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.48 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.49 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.50 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.51 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.292.2.52 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.292.2.53 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.292.2.54 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.292.2.55 mulhi\_fast()**

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]

```

**16.292.2.56 mod()**

```

INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static]

```

**16.292.2.57 signbits()**

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]

```

**16.292.2.58 zero()**

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

**16.292.3 Field Documentation****16.292.3.1 vect\_size**

```

constexpr const size_t vect_size = 4 [static], [constexpr]

```

**16.292.3.2 alignment**

```

constexpr const size_t alignment = 32 [static], [constexpr]

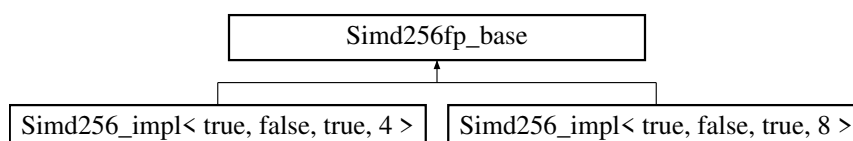
```

The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

**16.293 Simd256fp\_base Struct Reference**

Inheritance diagram for Simd256fp\_base:

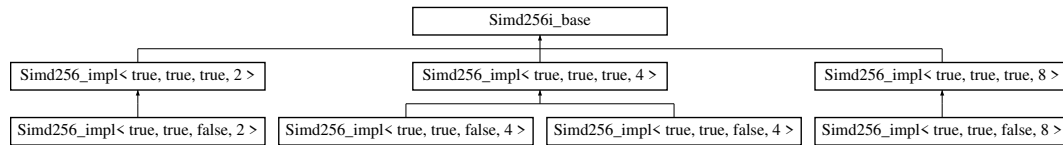


The documentation for this struct was generated from the following file:

- [simd256.inl](#)

## 16.294 Simd256i\_base Struct Reference

Inheritance diagram for Simd256i\_base:



### Public Types

- using [vect\\_t](#) = \_\_m256i

### Static Public Member Functions

- static [INLINE CONST vect\\_t zero](#) ()

### 16.294.1 Member Typedef Documentation

#### 16.294.1.1 vect\_t

using [vect\\_t](#) = \_\_m256i

### 16.294.2 Member Function Documentation

#### 16.294.2.1 zero()

static [INLINE CONST vect\\_t zero](#) ( ) [inline], [static]

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

## 16.295 Simd512\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

## 16.296 Simd512\_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- [simd512\\_float.inl](#)

## 16.297 Simd512\_impl< true, false, true, 8 > Struct Reference

### Public Types

- using [vect\\_t](#) = \_\_m512d
- using [scalar\\_t](#) = double
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>

- using `aligned_vector` = `std::vector< scalar_t, aligned_allocator >`
- `template<class Field >`  
using `is_same_element` = `std::is_same< typename Field::Element, scalar_t >`

## Static Public Member Functions

- static const `std::string` `type_string` ()
- `template<class T >`  
static constexpr bool `valid` (T \*p)
- `template<class T >`  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t` `zero` ()
- static `INLINE CONST vect_t` `set1` (const `scalar_t` x)
- static `INLINE CONST vect_t` `set` (const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8)
- `template<class T >`  
static `INLINE PURE vect_t` `gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t` `load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t` `loadu` (const `scalar_t` \*const p)
- static `INLINE` void `store` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE` void `storeu` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE` void `stream` (const `scalar_t` \*p, const `vect_t` v)
- `template<uint8_t s>`  
static `INLINE CONST vect_t` `shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t` `unpacklo_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `unpackhi_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE` void `unpacklohi` (`vect_t` &lo, `vect_t` &hi, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `pack_even` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE` void `pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE` void `transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3, `vect_t` &r4, `vect_t` &r5, `vect_t` &r6, `vect_t` &r7)
- `template<uint8_t s>`  
static `INLINE CONST vect_t` `blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `blendv` (const `vect_t` a, const `vect_t` b, const `vect_t` mask)
- static `INLINE CONST vect_t` `add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t` `addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t` `sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t` `mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `mulin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t` `div` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t` `greater_eq` (const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t floor` (const `vect_t` a)
- static `INLINE CONST vect_t ceil` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t hadd` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 64

## 16.297.1 Member Typedef Documentation

### 16.297.1.1 vect\_t

```
using vect_t = __m512d
```

### 16.297.1.2 scalar\_t

```
using scalar_t = double
```

### 16.297.1.3 aligned\_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

### 16.297.1.4 aligned\_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

### 16.297.1.5 is\_same\_element

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

## 16.297.2 Member Function Documentation

### 16.297.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.297.2.2 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.297.2.3 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

#### 16.297.2.4 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

#### 16.297.2.5 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

#### 16.297.2.6 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8 ) [inline], [static]
```

#### 16.297.2.7 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.297.2.8 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.297.2.9 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.297.2.10 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.297.2.11 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.297.2.12 stream()**

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.297.2.13 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.297.2.14 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.15 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.16 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.17 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.18 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.19 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.20 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.21 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.22 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static]
```

**16.297.2.23 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.24 blendv()**

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

**16.297.2.25 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.26 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.27 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.28 subin()**

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.29 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.30 mulin()**

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.31 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.32 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.33 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.34 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.35 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.36 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.37 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.38 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.39 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.40 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.41 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.42 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.297.2.43 floor()**

```
static INLINE CONST vect_t floor (
    const vect_t a ) [inline], [static]
```

**16.297.2.44 ceil()**

```
static INLINE CONST vect_t ceil (
    const vect_t a ) [inline], [static]
```

**16.297.2.45 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.297.2.46 hadd()**

```
static INLINE CONST vect_t hadd (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.297.2.47 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.297.3 Field Documentation****16.297.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

**16.297.3.2 alignment**

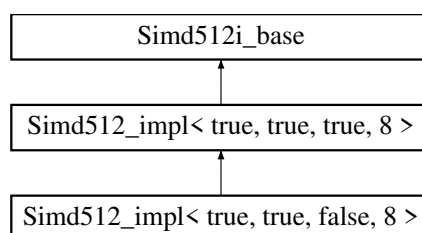
```
constexpr const size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_double.inl](#)

**16.298 Simd512\_impl< true, true, false, 8 > Struct Reference**

Inheritance diagram for Simd512\_impl< true, true, false, 8 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = uint64\_t
- using [aligned\\_allocator](#) = AlignedAllocator< [scalar\\_t](#), Alignment([alignment](#))>
- using [aligned\\_vector](#) = std::vector< [scalar\\_t](#), [aligned\\_allocator](#) >
- template<class Field >  
using [is\\_same\\_element](#) = std::is\_same< typename Field::Element, [scalar\\_t](#) >
- using [simdHalf](#) = Simd256< [scalar\\_t](#) >
- using [vect\\_t](#) = \_\_m512i
- using [half\\_t](#) = \_\_m256i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- template<uint8\_t k>  
static [INLINE](#) void [maskstore](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)

- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)`
- `static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)`
- `static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`
- `static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- `static INLINE vect_t mod (vect_t &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)`
- `static INLINE CONST vect_t zero ()`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static constexpr const size_t vect_size = 8`
- `static constexpr const size_t alignment = 64`

## Static Protected Member Functions

- `static INLINE CONST vect_t signbits (const vect_t x)`

### 16.298.1 Member Typedef Documentation

#### 16.298.1.1 scalar\_t

using `scalar_t` = `uint64_t`

**16.298.1.2 aligned\_allocator**

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

**16.298.1.3 aligned\_vector**

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.298.1.4 is\_same\_element**

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

**16.298.1.5 simdHalf**

```
using simdHalf = Simd256<scalar_t>
```

**16.298.1.6 vect\_t**

```
using vect_t = __m512i [inherited]
```

**16.298.1.7 half\_t**

```
using half_t = __m256i [inherited]
```

**16.298.2 Member Function Documentation****16.298.2.1 type\_string()**

```
static const std::string type_string ( ) [inline], [static]
```

**16.298.2.2 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.298.2.3 set() [1/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

#### 16.298.2.4 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.298.2.5 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.298.2.6 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.298.2.7 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.298.2.8 maskstore()

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.298.2.9 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.298.2.10 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.298.2.11 sra()

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

#### 16.298.2.12 greater()

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.298.2.13 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.298.2.14 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.15 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.16 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.298.2.17 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.298.2.18 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.19 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.20 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.21 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.22 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.23 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.24 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.298.2.25 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.298.2.26 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.298.2.27 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.298.2.28 set() [2/2]**

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static], [inherited]
```

**16.298.2.29 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.298.2.30 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.298.2.31 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.298.2.32 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.33 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.34 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.35 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.36 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.37 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.38 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.39 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.40 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static], [inherited]
```

**16.298.2.41 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.42 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.43 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.44 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.45 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.46 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.47 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.48 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.49 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.50 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.51 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.52 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.53 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.54 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.298.2.55 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.298.2.56 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (  
    vect_t x,  
    vect_t y ) [static], [inherited]
```

**16.298.2.57 mod()**

```
INLINE vect_t mod (  
    vect_t & C,  
    const __m512d & P,  
    const __m512d & INV_P,  
    const __m512d & NEG_P,  
    const vect_t & POW50REM,  
    const __m512d & MIN,  
    const __m512d & MAX,  
    __m512d & Q,  
    __m512d & T ) [static], [inherited]
```

**16.298.2.58 signbits()**

```
static INLINE CONST vect_t signbits (  
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.298.2.59 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.298.2.60 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.61 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.62 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.2.63 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.298.3 Field Documentation****16.298.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.298.3.2 alignment**

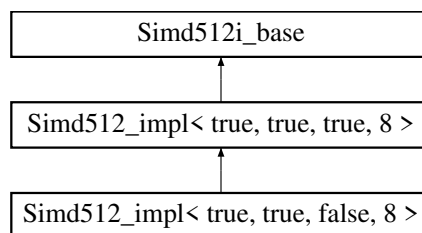
```
constexpr const size_t alignment = 64 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

**16.299 Simd512\_impl< true, true, true, 8 > Struct Reference**

Inheritance diagram for Simd512\_impl< true, true, true, 8 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m512i
- using [half\\_t](#) = \_\_m256i
- using [scalar\\_t](#) = int64\_t
- using [simdHalf](#) = [Simd256](#)< [scalar\\_t](#) >

- using `aligned_allocator` = `AlignedAllocator< scalar_t, Alignment(alignment)>`
- using `aligned_vector` = `std::vector< scalar_t, aligned_allocator >`
- template<class Field >  
using `is_same_element` = `std::is_same< typename Field::Element, scalar_t >`

## Static Public Member Functions

- static const std::string `type_string` ()
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- template<uint8\_t k>  
static `INLINE void maskstore` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_intrinsic` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &lo, `vect_t` &hi, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_even` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t pack_odd` (const `vect_t` a, const `vect_t` b)
- static `INLINE void pack` (`vect_t` &even, `vect_t` &odd, const `vect_t` a, const `vect_t` b)
- static `INLINE void transpose` (`vect_t` &r0, `vect_t` &r1, `vect_t` &r2, `vect_t` &r3, `vect_t` &r4, `vect_t` &r5, `vect_t` &r6, `vect_t` &r7)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)

- static `INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t mask_high ()`
- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)`
- static `INLINE CONST vect_t zero ()`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 64

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

## 16.299.1 Member Typedef Documentation

### 16.299.1.1 vect\_t

```
using vect_t = __m512i
```

### 16.299.1.2 half\_t

```
using half_t = __m256i
```

### 16.299.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.299.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

**16.299.1.5 aligned\_allocator**

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

**16.299.1.6 aligned\_vector**

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

**16.299.1.7 is\_same\_element**

```
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

**16.299.2 Member Function Documentation****16.299.2.1 type\_string()**

```
static const std::string type_string ( ) [inline], [static]
```

**16.299.2.2 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.299.2.3 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.299.2.4 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.299.2.5 set() [1/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.299.2.6 set() [2/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
```

```
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

#### 16.299.2.7 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.299.2.8 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.299.2.9 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.299.2.10 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.299.2.11 maskstore()

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.299.2.12 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.299.2.13 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.299.2.14 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.299.2.15 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.299.2.16 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.299.2.17 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.299.2.18 unpacklo\_intrinsic()**

```
static INLINE CONST vect_t unpacklo_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.19 unpackhi\_intrinsic()**

```
static INLINE CONST vect_t unpackhi_intrinsic (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.20 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.21 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.22 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & lo,  
    vect_t & hi,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.23 pack\_even()**

```
static INLINE CONST vect_t pack_even (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.24 pack\_odd()**

```
static INLINE CONST vect_t pack_odd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.25 pack()**

```
static INLINE void pack (  
    vect_t & even,  
    vect_t & odd,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.26 transpose()**

```
static INLINE void transpose (  
    vect_t & r0,  
    vect_t & r1,  
    vect_t & r2,  
    vect_t & r3,  
    vect_t & r4,  
    vect_t & r5,  
    vect_t & r6,  
    vect_t & r7 ) [inline], [static]
```

**16.299.2.27 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.28 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.29 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.30 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.31 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.32 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.299.2.33 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.34 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.299.2.35 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.36 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.37 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.38 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.39 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.40 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.41 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.42 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.43 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.44 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.45 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.299.2.46 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.299.2.47 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.48 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.49 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.50 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.51 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.52 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.299.2.53 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.299.2.54 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.299.2.55 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.299.2.56 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

**16.299.2.57 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INVP,
    const __m512d & NEGP,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static]
```

**16.299.2.58 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

**16.299.2.59 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.299.2.60 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.299.2.61 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.299.2.62 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

### 16.299.2.63 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

## 16.299.3 Field Documentation

### 16.299.3.1 vect\_size

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

### 16.299.3.2 alignment

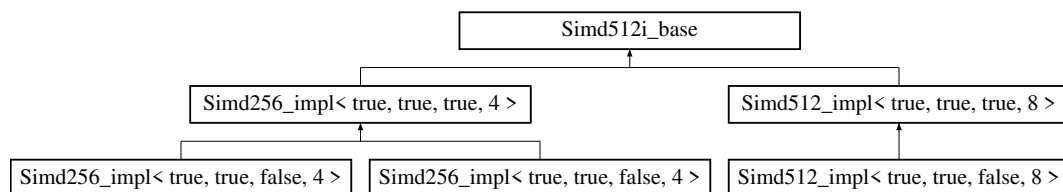
```
constexpr const size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

## 16.300 Simd512i\_base Struct Reference

Inheritance diagram for Simd512i\_base:



## Public Types

- using [vect\\_t](#) = \_\_m512i

## Static Public Member Functions

- static **INLINE** **CONST** [vect\\_t](#) zero ()
- static **INLINE** **CONST** [vect\\_t](#) vor (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static **INLINE** **CONST** [vect\\_t](#) vxor (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static **INLINE** **CONST** [vect\\_t](#) vand (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static **INLINE** **CONST** [vect\\_t](#) vandnot (const [vect\\_t](#) a, const [vect\\_t](#) b)

## 16.300.1 Member Typedef Documentation

### 16.300.1.1 vect\_t

```
using vect\_t = __m512i
```

## 16.300.2 Member Function Documentation

**16.300.2.1 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

**16.300.2.2 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.300.2.3 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.300.2.4 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.300.2.5 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

**16.301 SimdChooser< T, bool, bool > Struct Template Reference**

```
#include <fflas_simd.h>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

**16.302 SimdChooser< T, false, b > Struct Template Reference**

```
#include <fflas_simd.h>
```

**Public Types**

- using [value](#) = [NoSimd](#)< T >

**16.302.1 Member Typedef Documentation****16.302.1.1 value**

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.303 SimdChooser< T, true, false > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [value](#) = [NoSimd](#)< T >

### 16.303.1 Member Typedef Documentation

#### 16.303.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.304 SimdChooser< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [value](#) = [NoSimd](#)< T >

### 16.304.1 Member Typedef Documentation

#### 16.304.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.305 simdToType< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.306 Single Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.307 Sparse< Field, SparseMatrix\_t, IdxT, PtrT > Struct Template Reference

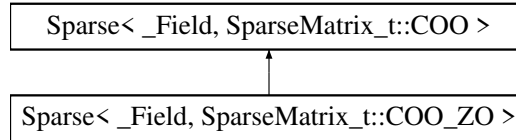
The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.308 Sparse< \_Field, SparseMatrix\_t::COO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::COO >:



### Public Types

- using [Field](#) = \_Field

### Data Fields

- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0

## 16.308.1 Member Typedef Documentation

### 16.308.1.1 Field

```
using Field = _Field
```

## 16.308.2 Field Documentation

### 16.308.2.1 col

```
index\_t* col = nullptr
```

### 16.308.2.2 row

```
index\_t* row = nullptr
```

### 16.308.2.3 dat

```
\_Field::Element\_ptr dat
```

**16.308.2.4 delayed**

```
bool delayed = false
```

**16.308.2.5 kmax**

```
uint64_t kmax = 0
```

**16.308.2.6 m**

```
index_t m = 0
```

**16.308.2.7 n**

```
index_t n = 0
```

**16.308.2.8 nnz**

```
uint64_t nnz = 0
```

**16.308.2.9 nElements**

```
uint64_t nElements = 0
```

**16.308.2.10 maxrow**

```
uint64_t maxrow = 0
```

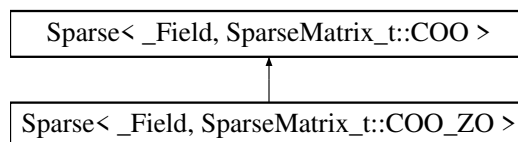
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.309 Sparse< \_Field, SparseMatrix\_t::COO\_ZO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::COO\_ZO >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- \_Field::Element [cst](#) = 1
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr

- `_Field::Element_ptr dat`
- `bool delayed = false`
- `uint64_t kmax = 0`
- `index_t m = 0`
- `index_t n = 0`
- `uint64_t nnz = 0`
- `uint64_t nElements = 0`
- `uint64_t maxrow = 0`

## 16.309.1 Member Typedef Documentation

### 16.309.1.1 Field

using `Field` = `_Field`

## 16.309.2 Field Documentation

### 16.309.2.1 cst

`_Field::Element cst = 1`

### 16.309.2.2 col

`index_t* col = nullptr` [inherited]

### 16.309.2.3 row

`index_t* row = nullptr` [inherited]

### 16.309.2.4 dat

`_Field::Element_ptr dat` [inherited]

### 16.309.2.5 delayed

`bool delayed = false` [inherited]

### 16.309.2.6 kmax

`uint64_t kmax = 0` [inherited]

### 16.309.2.7 m

`index_t m = 0` [inherited]

### 16.309.2.8 n

`index_t n = 0` [inherited]

**16.309.2.9 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.309.2.10 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.309.2.11 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

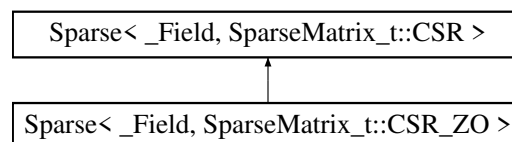
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.310 Sparse< \_Field, SparseMatrix\_t::CSR > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::CSR >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- uint64\_t [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- uint64\_t [nnz](#) = 0
- uint64\_t [nElements](#) = 0
- uint64\_t [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [index\\_t](#) \* [stend](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)

**16.310.1 Member Typedef Documentation****16.310.1.1 Field**

```
using Field = _Field
```

## 16.310.2 Field Documentation

### 16.310.2.1 delayed

```
bool delayed = false
```

### 16.310.2.2 kmax

```
uint64_t kmax = 0
```

### 16.310.2.3 m

```
index_t m = 0
```

### 16.310.2.4 n

```
index_t n = 0
```

### 16.310.2.5 nnz

```
uint64_t nnz = 0
```

### 16.310.2.6 nElements

```
uint64_t nElements = 0
```

### 16.310.2.7 maxrow

```
uint64_t maxrow = 0
```

### 16.310.2.8 col

```
index_t* col = nullptr
```

### 16.310.2.9 st

```
index_t* st = nullptr
```

### 16.310.2.10 stend

```
index_t* stend = nullptr
```

### 16.310.2.11 dat

```
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.311 Sparse< \_Field, SparseMatrix\_t::CSR\_HYB > Struct Template Reference

```
#include <csr_hyb.h>
```

### Public Types

- using [Field](#) = \_Field

### Data Fields

- bool [delayed](#) = false
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [uint64\\_t](#) [nOnes](#) = 0
- [uint64\\_t](#) [nMOnes](#) = 0
- [uint64\\_t](#) [nOthers](#) = 0

### 16.311.1 Member Typedef Documentation

#### 16.311.1.1 Field

```
using Field = _Field
```

### 16.311.2 Field Documentation

#### 16.311.2.1 delayed

```
bool delayed = false
```

#### 16.311.2.2 col

```
index\_t* col = nullptr
```

#### 16.311.2.3 st

```
index\_t* st = nullptr
```

#### 16.311.2.4 dat

```
\_Field::Element\_ptr dat
```

**16.311.2.5 kmax**

```
uint64_t kmax = 0
```

**16.311.2.6 m**

```
index_t m = 0
```

**16.311.2.7 n**

```
index_t n = 0
```

**16.311.2.8 nnz**

```
uint64_t nnz = 0
```

**16.311.2.9 nElements**

```
uint64_t nElements = 0
```

**16.311.2.10 maxrow**

```
uint64_t maxrow = 0
```

**16.311.2.11 nOnes**

```
uint64_t nOnes = 0
```

**16.311.2.12 nMOnes**

```
uint64_t nMOnes = 0
```

**16.311.2.13 nOthers**

```
uint64_t nOthers = 0
```

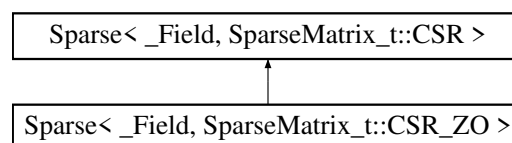
The documentation for this struct was generated from the following file:

- [csr\\_hyb.h](#)

## 16.312 Sparse<\_Field, SparseMatrix\_t::CSR\_ZO > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::CSR\_ZO >:



## Public Types

- using [Field](#) = \_Field

## Data Fields

- `int64_t` [cst](#) = 1
- `bool` [delayed](#) = false
- `uint64_t` [kmax](#) = 0
- `index_t` [m](#) = 0
- `index_t` [n](#) = 0
- `uint64_t` [nnz](#) = 0
- `uint64_t` [nElements](#) = 0
- `uint64_t` [maxrow](#) = 0
- `index_t` \* [col](#) = nullptr
- `index_t` \* [st](#) = nullptr
- `index_t` \* [stend](#) = nullptr
- `_Field::Element_ptr` [dat](#)

## 16.312.1 Member Typedef Documentation

### 16.312.1.1 Field

using [Field](#) = \_Field

## 16.312.2 Field Documentation

### 16.312.2.1 cst

`int64_t` [cst](#) = 1

### 16.312.2.2 delayed

`bool` [delayed](#) = false

### 16.312.2.3 kmax

`uint64_t` [kmax](#) = 0 [inherited]

### 16.312.2.4 m

`index_t` [m](#) = 0 [inherited]

### 16.312.2.5 n

`index_t` [n](#) = 0 [inherited]

### 16.312.2.6 nnz

`uint64_t` [nnz](#) = 0 [inherited]

**16.312.2.7 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.312.2.8 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.312.2.9 col**

```
index_t* col = nullptr [inherited]
```

**16.312.2.10 st**

```
index_t* st = nullptr [inherited]
```

**16.312.2.11 stend**

```
index_t* stend = nullptr [inherited]
```

**16.312.2.12 dat**

```
_Field::Element_ptr dat [inherited]
```

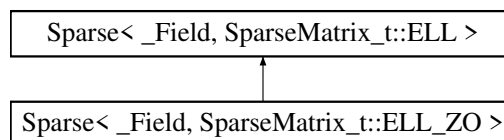
The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.313 Sparse<\_Field, SparseMatrix\_t::ELL > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::ELL >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- uint64\_t [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- uint64\_t [nnz](#) = 0
- uint64\_t [nElements](#) = 0

- `uint64_t maxrow = 0`
- `index_t * col = nullptr`
- `_Field::Element_ptr dat`

## 16.313.1 Member Typedef Documentation

### 16.313.1.1 Field

using `Field` = `_Field`

## 16.313.2 Field Documentation

### 16.313.2.1 delayed

`bool delayed = false`

### 16.313.2.2 kmax

`uint64_t kmax = 0`

### 16.313.2.3 m

`index_t m = 0`

### 16.313.2.4 n

`index_t n = 0`

### 16.313.2.5 ld

`index_t ld = 0`

### 16.313.2.6 nnz

`uint64_t nnz = 0`

### 16.313.2.7 nElements

`uint64_t nElements = 0`

### 16.313.2.8 maxrow

`uint64_t maxrow = 0`

### 16.313.2.9 col

`index_t* col = nullptr`

### 16.313.2.10 dat

`_Field::Element_ptr` dat

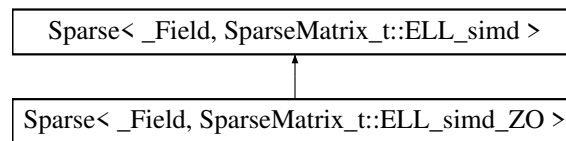
The documentation for this struct was generated from the following file:

- `ell.h`

## 16.314 Sparse< \_Field, SparseMatrix\_t::ELL\_simd > Struct Template Reference

`#include <ell_simd.h>`

Inheritance diagram for `Sparse< _Field, SparseMatrix_t::ELL_simd >`:



### Data Fields

- bool `delayed` = false
- int `chunk` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `ld` = 0
- `uint64_t` `kmax` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `uint64_t` `nChunks` = 0
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` dat

### 16.314.1 Field Documentation

#### 16.314.1.1 delayed

`bool delayed = false`

#### 16.314.1.2 chunk

`int chunk = 0`

#### 16.314.1.3 m

`index_t m = 0`

#### 16.314.1.4 n

`index_t n = 0`

**16.314.1.5 ld**

```
index_t ld = 0
```

**16.314.1.6 kmax**

```
uint64_t kmax = 0
```

**16.314.1.7 nnz**

```
uint64_t nnz = 0
```

**16.314.1.8 nElements**

```
uint64_t nElements = 0
```

**16.314.1.9 maxrow**

```
uint64_t maxrow = 0
```

**16.314.1.10 nChunks**

```
uint64_t nChunks = 0
```

**16.314.1.11 col**

```
index_t* col = nullptr
```

**16.314.1.12 dat**

```
_Field::Element_ptr dat
```

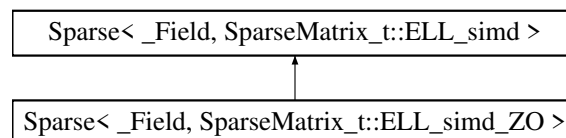
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.315 Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO > Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >:



## Data Fields

- `_Field::Element` `cst` = 1
- `bool` `delayed` = false
- `int` `chunk` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `ld` = 0
- `uint64_t` `kmax` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `uint64_t` `nChunks` = 0
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

### 16.315.1 Field Documentation

#### 16.315.1.1 `cst`

```
_Field::Element cst = 1
```

#### 16.315.1.2 `delayed`

```
bool delayed = false [inherited]
```

#### 16.315.1.3 `chunk`

```
int chunk = 0 [inherited]
```

#### 16.315.1.4 `m`

```
index_t m = 0 [inherited]
```

#### 16.315.1.5 `n`

```
index_t n = 0 [inherited]
```

#### 16.315.1.6 `ld`

```
index_t ld = 0 [inherited]
```

#### 16.315.1.7 `kmax`

```
uint64_t kmax = 0 [inherited]
```

#### 16.315.1.8 `nnz`

```
uint64_t nnz = 0 [inherited]
```

**16.315.1.9 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.315.1.10 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.315.1.11 nChunks**

```
uint64_t nChunks = 0 [inherited]
```

**16.315.1.12 col**

```
index_t* col = nullptr [inherited]
```

**16.315.1.13 dat**

```
_Field::Element_ptr dat [inherited]
```

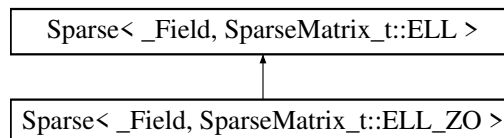
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.316 Sparse< \_Field, SparseMatrix\_t::ELL\_ZO > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- uint64\_t [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- uint64\_t [nnz](#) = 0
- uint64\_t [nElements](#) = 0
- uint64\_t [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

## 16.316.1 Member Typedef Documentation

### 16.316.1.1 Field

```
using Field = _Field
```

## 16.316.2 Field Documentation

### 16.316.2.1 cst

```
_Field::Element cst = 1
```

### 16.316.2.2 delayed

```
bool delayed = false [inherited]
```

### 16.316.2.3 kmax

```
uint64_t kmax = 0 [inherited]
```

### 16.316.2.4 m

```
index_t m = 0 [inherited]
```

### 16.316.2.5 n

```
index_t n = 0 [inherited]
```

### 16.316.2.6 ld

```
index_t ld = 0 [inherited]
```

### 16.316.2.7 nnz

```
uint64_t nnz = 0 [inherited]
```

### 16.316.2.8 nElements

```
uint64_t nElements = 0 [inherited]
```

### 16.316.2.9 maxrow

```
uint64_t maxrow = 0 [inherited]
```

### 16.316.2.10 col

```
index_t* col = nullptr [inherited]
```

**16.316.2.11 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [ell.h](#)

**16.317 Sparse<\_Field, SparseMatrix\_t::HYB\_ZO > Struct Template Reference**

```
#include <hyb_zo.h>
```

**Public Types**

- using [Field](#) = [\\_Field](#)
- typedef [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > [Self\\_t](#)

**Data Fields**

- bool [delayed](#) = false
- uint64\_t [kmax](#) = 0
- [index\\_t](#) m = 0
- [index\\_t](#) n = 0
- uint64\_t [nnz](#) = 0
- uint64\_t [maxrow](#) = 0
- uint64\_t [nElements](#) = 0
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR](#) > \* [dat](#) = nullptr
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > \* [one](#) = nullptr
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > \* [mone](#) = nullptr

**16.317.1 Member Typedef Documentation****16.317.1.1 Field**

```
using Field = \_Field
```

**16.317.1.2 Self\_t**

```
typedef Sparse< \_Field, SparseMatrix\_t::HYB\_ZO > Self\_t
```

**16.317.2 Field Documentation****16.317.2.1 delayed**

```
bool delayed = false
```

**16.317.2.2 kmax**

```
uint64_t kmax = 0
```

**16.317.2.3 m**

```
index_t m = 0
```

**16.317.2.4 n**

```
index_t n = 0
```

**16.317.2.5 nnz**

```
uint64_t nnz = 0
```

**16.317.2.6 maxrow**

```
uint64_t maxrow = 0
```

**16.317.2.7 nElements**

```
uint64_t nElements = 0
```

**16.317.2.8 dat**

```
Sparse<_Field, SparseMatrix_t::CSR>* dat = nullptr
```

**16.317.2.9 one**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* one = nullptr
```

**16.317.2.10 mone**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* mone = nullptr
```

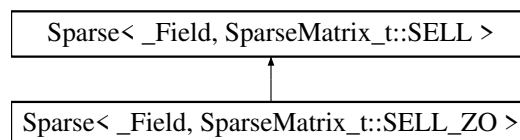
The documentation for this struct was generated from the following file:

- [hyb\\_zo.h](#)

## 16.318 Sparse<\_Field, SparseMatrix\_t::SELL > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::SELL >:



### Public Types

- using [Field](#) = \_Field

## Data Fields

- bool `delayed` = false
- int `chunk` = 0
- `index_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `maxrow` = 0
- `index_t` `sigma` = 0
- `index_t` `nChunks` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `index_t` \* `perm` = nullptr
- `uint64_t` \* `st` = nullptr
- `index_t` \* `chunkSize` = nullptr
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

## 16.318.1 Member Typedef Documentation

### 16.318.1.1 Field

using `Field` = `_Field`

## 16.318.2 Field Documentation

### 16.318.2.1 delayed

`bool` `delayed` = false

### 16.318.2.2 chunk

`int` `chunk` = 0

### 16.318.2.3 kmax

`index_t` `kmax` = 0

### 16.318.2.4 m

`index_t` `m` = 0

### 16.318.2.5 n

`index_t` `n` = 0

### 16.318.2.6 maxrow

`index_t` `maxrow` = 0

**16.318.2.7 sigma**

```
index_t sigma = 0
```

**16.318.2.8 nChunks**

```
index_t nChunks = 0
```

**16.318.2.9 nnz**

```
uint64_t nnz = 0
```

**16.318.2.10 nElements**

```
uint64_t nElements = 0
```

**16.318.2.11 perm**

```
index_t* perm = nullptr
```

**16.318.2.12 st**

```
uint64_t* st = nullptr
```

**16.318.2.13 chunkSize**

```
index_t* chunkSize = nullptr
```

**16.318.2.14 col**

```
index_t* col = nullptr
```

**16.318.2.15 dat**

```
_Field::Element_ptr dat
```

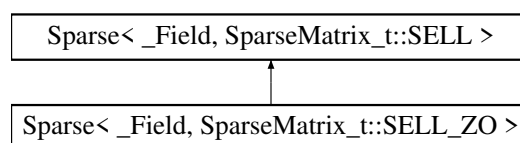
The documentation for this struct was generated from the following file:

- [sell.h](#)

## 16.319 Sparse<\_Field, SparseMatrix\_t::SELL\_ZO > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::SELL\_ZO >:



## Public Types

- using [Field](#) = \_Field

## Data Fields

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- int [chunk](#) = 0
- [index\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [maxrow](#) = 0
- [index\\_t](#) [sigma](#) = 0
- [index\\_t](#) [nChunks](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [index\\_t](#) \* [perm](#) = nullptr
- [uint64\\_t](#) \* [st](#) = nullptr
- [index\\_t](#) \* [chunkSize](#) = nullptr
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

## 16.319.1 Member Typedef Documentation

### 16.319.1.1 Field

using [Field](#) = \_Field

## 16.319.2 Field Documentation

### 16.319.2.1 cst

[\\_Field::Element](#) [cst](#) = 1

### 16.319.2.2 delayed

bool [delayed](#) = false [inherited]

### 16.319.2.3 chunk

int [chunk](#) = 0 [inherited]

### 16.319.2.4 kmax

[index\\_t](#) [kmax](#) = 0 [inherited]

### 16.319.2.5 m

[index\\_t](#) [m](#) = 0 [inherited]

**16.319.2.6 n**

```
index_t n = 0 [inherited]
```

**16.319.2.7 maxrow**

```
index_t maxrow = 0 [inherited]
```

**16.319.2.8 sigma**

```
index_t sigma = 0 [inherited]
```

**16.319.2.9 nChunks**

```
index_t nChunks = 0 [inherited]
```

**16.319.2.10 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.319.2.11 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.319.2.12 perm**

```
index_t* perm = nullptr [inherited]
```

**16.319.2.13 st**

```
uint64_t* st = nullptr [inherited]
```

**16.319.2.14 chunkSize**

```
index_t* chunkSize = nullptr [inherited]
```

**16.319.2.15 col**

```
index_t* col = nullptr [inherited]
```

**16.319.2.16 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [sell.h](#)

**16.320 SpMat< Field, flag > Struct Template Reference**

```
#include <fflas_sparse.h>
```

## Data Fields

- [FFLAS::CooMat<Field> \\* \\_coo](#) = nullptr
- [FFLAS::CsrMat<Field> \\* \\_csr](#) = nullptr
- [FFLAS::EllMat<Field> \\* \\_ell](#) = nullptr

### 16.320.1 Field Documentation

#### 16.320.1.1 \_coo

```
FFLAS::CooMat<Field>* _coo = nullptr
```

#### 16.320.1.2 \_csr

```
FFLAS::CsrMat<Field>* _csr = nullptr
```

#### 16.320.1.3 \_ell

```
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.321 StatsMatrix Struct Reference

```
#include <utils.h>
```

## Data Fields

- uint64\_t [rowdim](#) = 0
- uint64\_t [coldim](#) = 0
- uint64\_t [nOnes](#) = 0
- uint64\_t [nMOnes](#) = 0
- uint64\_t [nOthers](#) = 0
- uint64\_t [nnz](#) = 0
- uint64\_t [maxRow](#) = 0
- uint64\_t [minRow](#) = 0
- uint64\_t [averageRow](#) = 0
- uint64\_t [deviationRow](#) = 0
- uint64\_t [maxCol](#) = 0
- uint64\_t [minCol](#) = 0
- uint64\_t [averageCol](#) = 0
- uint64\_t [deviationCol](#) = 0
- uint64\_t [minColDifference](#) = 0
- uint64\_t [maxColDifference](#) = 0
- uint64\_t [averageColDifference](#) = 0
- uint64\_t [deviationColDifference](#) = 0
- uint64\_t [minRowDifference](#) = 0
- uint64\_t [maxRowDifference](#) = 0
- uint64\_t [averageRowDifference](#) = 0
- uint64\_t [deviationRowDifference](#) = 0
- uint64\_t [nDenseRows](#) = 0
- uint64\_t [nDenseCols](#) = 0

- `uint64_t nEmptyRows = 0`
- `uint64_t nEmptyCols = 0`
- `uint64_t nEmptyColsEnd = 0`
- `std::vector< uint64_t > denseRows`
- `std::vector< uint64_t > denseCols`

## 16.321.1 Field Documentation

### 16.321.1.1 rowdim

`uint64_t rowdim = 0`

### 16.321.1.2 coldim

`uint64_t coldim = 0`

### 16.321.1.3 nOnes

`uint64_t nOnes = 0`

### 16.321.1.4 nMOnes

`uint64_t nMOnes = 0`

### 16.321.1.5 nOthers

`uint64_t nOthers = 0`

### 16.321.1.6 nnz

`uint64_t nnz = 0`

### 16.321.1.7 maxRow

`uint64_t maxRow = 0`

### 16.321.1.8 minRow

`uint64_t minRow = 0`

### 16.321.1.9 averageRow

`uint64_t averageRow = 0`

### 16.321.1.10 deviationRow

`uint64_t deviationRow = 0`

**16.321.1.11 maxCol**

```
uint64_t maxCol = 0
```

**16.321.1.12 minCol**

```
uint64_t minCol = 0
```

**16.321.1.13 averageCol**

```
uint64_t averageCol = 0
```

**16.321.1.14 deviationCol**

```
uint64_t deviationCol = 0
```

**16.321.1.15 minColDifference**

```
uint64_t minColDifference = 0
```

**16.321.1.16 maxColDifference**

```
uint64_t maxColDifference = 0
```

**16.321.1.17 averageColDifference**

```
uint64_t averageColDifference = 0
```

**16.321.1.18 deviationColDifference**

```
uint64_t deviationColDifference = 0
```

**16.321.1.19 minRowDifference**

```
uint64_t minRowDifference = 0
```

**16.321.1.20 maxRowDifference**

```
uint64_t maxRowDifference = 0
```

**16.321.1.21 averageRowDifference**

```
uint64_t averageRowDifference = 0
```

**16.321.1.22 deviationRowDifference**

```
uint64_t deviationRowDifference = 0
```

**16.321.1.23 nDenseRows**

```
uint64_t nDenseRows = 0
```

**16.321.1.24 nDenseCols**

```
uint64_t nDenseCols = 0
```

**16.321.1.25 nEmptyRows**

```
uint64_t nEmptyRows = 0
```

**16.321.1.26 nEmptyCols**

```
uint64_t nEmptyCols = 0
```

**16.321.1.27 nEmptyColsEnd**

```
uint64_t nEmptyColsEnd = 0
```

**16.321.1.28 denseRows**

```
std::vector<uint64_t> denseRows
```

**16.321.1.29 denseCols**

```
std::vector<uint64_t> denseCols
```

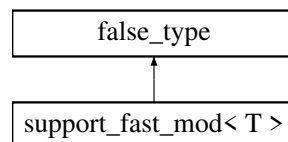
The documentation for this struct was generated from the following file:

- [utils.h](#)

**16.322 support\_fast\_mod< T > Struct Template Reference**

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< T >:



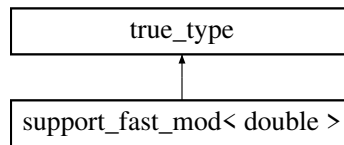
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

**16.323 support\_fast\_mod< double > Struct Reference**

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< double >:



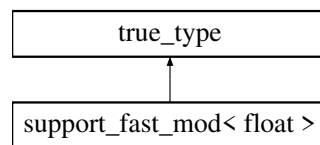
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.324 support\_fast\_mod< float > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< float >:



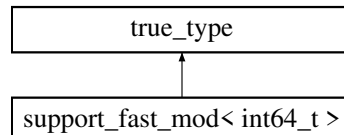
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.325 support\_fast\_mod< int64\_t > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< int64\_t >:



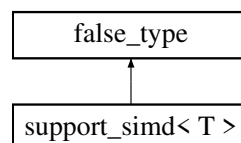
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.326 support\_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Inheritance diagram for support\_simd< T >:



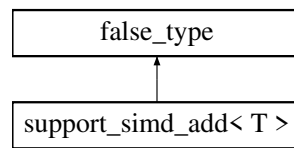
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.327 support\_simd\_add< T > Struct Template Reference

```
#include <fflas_fadd.h>
```

Inheritance diagram for support\_simd\_add< T >:



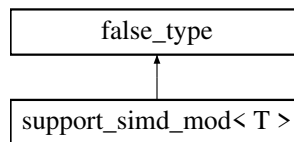
The documentation for this struct was generated from the following file:

- [fflas\\_fadd.h](#)

## 16.328 support\_simd\_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_simd\_mod< T >:



The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.329 Test< Elt > Class Template Reference

### Public Types

- using [Field](#) = Modular< Elt >
- using [Elt\\_ptr](#) = typename [Field::Element\\_ptr](#)
- using [Residu](#) = typename [Field::Residu\\_t](#)
- template<bool B, class T = void>  
using [enable\\_if\\_t](#) = typename std::enable\_if< B, T >::type
- template<typename Simd >  
using [is\\_same\\_element](#) = typename Simd::template [is\\_same\\_element](#)< [Field](#) >
- template<typename E >  
using [enable\\_if\\_no\\_simd\\_t](#) = [enable\\_if\\_t](#)< [Simd](#)< E >::vect\_size==1 >
- template<typename E >  
using [enable\\_if\\_simd128\\_t](#) = [enable\\_if\\_t](#)< sizeof(E) \* [Simd](#)< E >::vect\_size==16 >
- template<typename E >  
using [enable\\_if\\_simd256\\_t](#) = [enable\\_if\\_t](#)< sizeof(E) \* [Simd](#)< E >::vect\_size==32 >
- template<typename E >  
using [enable\\_if\\_simd512\\_t](#) = [enable\\_if\\_t](#)< sizeof(E) \* [Simd](#)< E >::vect\_size==64 >

### Public Member Functions

- [Test](#) (size\_t mm, size\_t nn)
- template<typename Simd = NoSimd<Elt>, enable\_if\_t< is\_same\_element< Simd >::value > \* = nullptr>  
bool [test\\_ftranspose](#) (size\_t m, size\_t n, [Elt\\_ptr](#) A, size\_t lda, [Elt\\_ptr](#) B, size\_t ldb)

- `template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * = nullptr>`  
`bool doTests ()`
- `template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t< _E > * = nullptr>`  
`bool run ()`

## Static Public Member Functions

- `template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>`  
`static Residu cardinality ()`
- `template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>`  
`static Residu cardinality ()`

## Protected Attributes

- `Field F`
- `size_t _mm`
- `size_t _nn`

## 16.329.1 Member Typedef Documentation

### 16.329.1.1 Field

`using Field = Modular<Elt>`

### 16.329.1.2 Elt\_ptr

`using Elt_ptr = typename Field::Element_ptr`

### 16.329.1.3 Residu

`using Residu = typename Field::Residu_t`

### 16.329.1.4 enable\_if\_t

`using enable_if_t = typename std::enable_if<B, T>::type`

### 16.329.1.5 is\_same\_element

`using is_same_element = typename Simd::template is_same_element<Field>`

### 16.329.1.6 enable\_if\_no\_simd\_t

`using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>`

### 16.329.1.7 enable\_if\_simd128\_t

`using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>`

**16.329.1.8 enable\_if\_simd256\_t**

```
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

**16.329.1.9 enable\_if\_simd512\_t**

```
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

**16.329.2 Constructor & Destructor Documentation****16.329.2.1 Test()**

```
Test (
    size_t mm,
    size_t nn ) [inline]
```

**16.329.3 Member Function Documentation****16.329.3.1 cardinality() [1/2]**

```
static Residu cardinality ( ) [inline], [static]
```

**16.329.3.2 cardinality() [2/2]**

```
static Residu cardinality ( ) [inline], [static]
```

**16.329.3.3 test\_ftranspose()**

```
bool test_ftranspose (
    size_t m,
    size_t n,
    Elt_ptr A,
    size_t lda,
    Elt_ptr B,
    size_t ldb ) [inline]
```

**16.329.3.4 doTests()**

```
bool doTests ( ) [inline]
```

**16.329.3.5 run()**

```
bool run ( ) [inline]
```

**16.329.4 Field Documentation**

## 16.329.4.1 F

Field F [protected]

## 16.329.4.2 \_mm

size\_t \_mm [protected]

## 16.329.4.3 \_nn

size\_t \_nn [protected]

The documentation for this class was generated from the following file:

- [test-storage-transpose.C](#)

## 16.330 TestOneMethod< Simd > Class Template Reference

### Public Types

- using [Element](#) = typename Simd::scalar\_t
- using [vect\\_t](#) = typename Simd::vect\_t
- using [vectElt](#) = vector< [Element](#) >
- template<bool B, typename T = void>  
using [enable\\_if\\_t](#) = typename enable\_if< B, T >::type

### Public Member Functions

- template<typename... AScal, typename RScal, typename... ASimd, typename RSimd, enable\_if\_t< sizeof...(AScal)==sizeof...(ASimd)> \* = nullptr, enable\_if\_t< count\_nonconst\_lvalue\_reference< AScal... >::n==count\_nonconst\_lvalue\_reference< ASimd... >::n > \* = nullptr, enable\_if\_t< is\_all\_same< AScal... >::value > \* = nullptr, enable\_if\_t< is\_all\_same< vect\_t, ASimd... >::value > \* = nullptr>  
[TestOneMethod](#) (function< RSimd(ASimd...)> fsimd, function< RScal(AScal...)> fscal, function< void(vector< [vectElt](#) > &)> genInputs, string fname)
- template<typename Ret, typename... AScal>  
[enable\\_if\\_t< is\\_all\\_same< Element, AScal... >::value &&std::is\\_convertible< Ret, Element >::value, void > evaluate\\_scalar\\_method](#) (function< Ret(AScal...)> fscal)
- template<typename... AScal>  
[enable\\_if\\_t< is\\_all\\_same< vectElt, AScal... >::value, void > evaluate\\_scalar\\_method](#) (function< [vectElt](#)(AScal...)> fscal)
- template<typename... AScal>  
[enable\\_if\\_t< is\\_all\\_same< vectElt, AScal... >::value, void > evaluate\\_scalar\\_method](#) (function< void(AScal...)> fscal)
- template<typename Ret, typename... ASimd>  
[enable\\_if\\_t< is\\_all\\_same< vect\\_t, ASimd... >::value &&std::is\\_convertible< Ret, vect\\_t >::value, void > evaluate\\_simd\\_method](#) (function< Ret(ASimd...)> fsimd, array< [vect\\_t](#), sizeof...(ASimd)> &simd\_in)
- template<typename... ASimd>  
[enable\\_if\\_t< is\\_all\\_same< vect\\_t, ASimd... >::value, void > evaluate\\_simd\\_method](#) (function< void(ASimd...)> fsimd, array< [vect\\_t](#), sizeof...(ASimd)> &simd\_in)
- bool [getStatus](#) () const
- string [getTestName](#) () const
- bool [writeResultLine](#) () const
- void [writeDebugData](#) () const

### Static Public Attributes

- constexpr static size\_t [vect\\_size](#) = Simd::vect\_size

## Protected Attributes

- size\_t [nb\\_lref](#)
- string [name](#)
- vector< [vectElt](#) > [inputs](#)
- vector< [vectElt](#) > [outputs\\_simd](#)
- vector< [vectElt](#) > [outputs\\_scalar](#)

## 16.330.1 Member Typedef Documentation

### 16.330.1.1 Element

```
using Element = typename Simd::scalar_t
```

### 16.330.1.2 vect\_t

```
using vect\_t = typename Simd::vect_t
```

### 16.330.1.3 vectElt

```
using vectElt = vector<Element>
```

### 16.330.1.4 enable\_if\_t

```
using enable\_if\_t = typename enable_if<B, T>::type
```

## 16.330.2 Constructor & Destructor Documentation

### 16.330.2.1 TestOneMethod()

```
TestOneMethod (
    function< RSimd(ASimd...)> fsimd,
    function< RScal(AScal...)> fscal,
    function< void(vector< vectElt > &)> genInputs,
    string fname ) [inline]
```

## 16.330.3 Member Function Documentation

### 16.330.3.1 evaluate\_scalar\_method() [1/3]

```
enable\_if\_t<is\_all\_same<Element, AScal...>::value && std::is_convertible<Ret, Element>↔
::value, void> evaluate\_scalar\_method (
    function< Ret(AScal...)> fscal ) [inline]
```

### 16.330.3.2 evaluate\_scalar\_method() [2/3]

```
enable\_if\_t<is\_all\_same<vectElt, AScal...>::value, void> evaluate\_scalar\_method (
    function< vectElt(AScal...)> fscal ) [inline]
```

**16.330.3.3 evaluate\_scalar\_method() [3/3]**

```
enable_if_t<is_all_same<vect_elt, AScal...>::value, void> evaluate_scalar_method (
    function< void(AScal...)> fscal ) [inline]
```

**16.330.3.4 evaluate\_simd\_method() [1/2]**

```
enable_if_t<is_all_same<vect_t, ASimd...>::value && std::is_convertible<Ret, vect_t>::value,
void> evaluate_simd_method (
    function< Ret(ASimd...)> fsimd,
    array< vect_t, sizeof...(ASimd)> & simd_in ) [inline]
```

**16.330.3.5 evaluate\_simd\_method() [2/2]**

```
enable_if_t<is_all_same<vect_t, ASimd...>::value, void> evaluate_simd_method (
    function< void(ASimd...)> fsimd,
    array< vect_t, sizeof...(ASimd)> & simd_in ) [inline]
```

**16.330.3.6 getStatus()**

```
bool getStatus ( ) const [inline]
```

**16.330.3.7 getTestName()**

```
string getTestName ( ) const [inline]
```

**16.330.3.8 writeResultLine()**

```
bool writeResultLine ( ) const [inline]
```

**16.330.3.9 writeDebugData()**

```
void writeDebugData ( ) const [inline]
```

**16.330.4 Field Documentation****16.330.4.1 vect\_size**

```
constexpr static size_t vect_size = Simd::vect_size [static], [constexpr]
```

**16.330.4.2 nb\_lref**

```
size_t nb_lref [protected]
```

**16.330.4.3 name**

```
string name [protected]
```

**16.330.4.4 inputs**

```
vector<vectElt> inputs [protected]
```

**16.330.4.5 outputs\_simd**

```
vector<vectElt> outputs_simd [protected]
```

**16.330.4.6 outputs\_scalar**

```
vector<vectElt> outputs_scalar [protected]
```

The documentation for this class was generated from the following file:

- [test-simd.C](#)

**16.331 tfn\_minus Struct Reference**

```
#include <sparse_matrix_traits.h>
```

**Public Member Functions**

- `template<typename... Args>`  
`auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

**16.331.1 Member Function Documentation****16.331.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.332 tfn\_minus\_eq Struct Reference**

```
#include <sparse_matrix_traits.h>
```

**Public Member Functions**

- `template<typename... Args>`  
`auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

**16.332.1 Member Function Documentation****16.332.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.333 tfn\_mul Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.333.1 Member Function Documentation

#### 16.333.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.334 tfn\_mul\_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.334.1 Member Function Documentation

#### 16.334.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.335 tfn\_plus Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.335.1 Member Function Documentation

**16.335.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.336 tfn\_plus\_eq Struct Reference**

```
#include <sparse_matrix_traits.h>
```

**Public Member Functions**

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

**16.336.1 Member Function Documentation****16.336.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.337 Threads Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.338 ThreeD Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.339 ThreeDAdaptive Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.340 ThreeDInPlace Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.341 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference

TRSM Helper.

### Public Member Functions

- `template<class Cut , class Param >`  
[TRSMHelper](#) ([ParSeqHelper::Parallel](#)< Cut, Param > \_PS)
- `TRSMHelper` ([ParSeqHelper::Sequential](#) \_PS)
- `template<typename RIT , typename PST >`  
[TRSMHelper](#) ([TRSMHelper](#)< RIT, PST > &\_TH)
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>`  
[FFLAS::MMHelper](#)< Dom, Algo, ModeT, ParSeqTrait > [pMMH](#) (Dom &D, size\_t m, size\_t k, size\_t n, ParSeqTrait p) const
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>`  
[FFLAS::MMHelper](#)< Dom, Algo, ModeT, ParSeqTrait > [pMMH](#) (Dom &D, size\_t m, size\_t k, size\_t n) const

### Data Fields

- ParSeqTrait [parseq](#)

### 16.341.1 Detailed Description

```
template<typename ReclterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< ReclterTrait, ParSeqTrait >
```

TRSM Helper.

### 16.341.2 Constructor & Destructor Documentation

#### 16.341.2.1 TRSMHelper() [1/3]

```
TRSMHelper (
    ParSeqHelper::Parallel< Cut, Param > _PS ) [inline]
```

#### 16.341.2.2 TRSMHelper() [2/3]

```
TRSMHelper (
    ParSeqHelper::Sequential _PS ) [inline]
```

#### 16.341.2.3 TRSMHelper() [3/3]

```
TRSMHelper (
    TRSMHelper< RIT, PST > &_TH ) [inline]
```

### 16.341.3 Member Function Documentation

**16.341.3.1 pMMH() [1/2]**

```
FFLAS::MMHelper<Dom, Algo, ModeT, ParSeqTrait> pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait p ) const [inline]
```

**16.341.3.2 pMMH() [2/2]**

```
FFLAS::MMHelper<Dom, Algo, ModeT, ParSeqTrait> pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n ) const [inline]
```

**16.341.4 Field Documentation****16.341.4.1 parseq**

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.342 TwoD Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.343 TwoDAdaptive Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.344 UnparametricTag Struct Reference**

If the field uses a representation with infix operators.

```
#include <field-traits.h>
```

**16.344.1 Detailed Description**

If the field uses a representation with infix operators.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.345 width< T > Struct Template Reference****Static Public Attributes**

- static constexpr size\_t [value](#) = 2+2\*sizeof(T)

### 16.345.1 Field Documentation

#### 16.345.1.1 value

```
constexpr size_t value = 2+2*sizeof(T) [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.346 width< double > Struct Reference

### Static Public Attributes

- static constexpr size\_t [value](#) = 24

### 16.346.1 Field Documentation

#### 16.346.1.1 value

```
constexpr size_t value = 24 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.347 width< float > Struct Reference

### Static Public Attributes

- static constexpr size\_t [value](#) = 16

### 16.347.1 Field Documentation

#### 16.347.1.1 value

```
constexpr size_t value = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.348 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.349 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)



## Chapter 17

# File Documentation

### 17.1 101-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.1.1 Function Documentation

##### 17.1.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.2 2x2-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.2.1 Function Documentation

#### 17.2.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.3 2x2-fftrsv.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
#include <array>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.3.1 Function Documentation

##### 17.3.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.4 2x2-pluq.C File Reference

```
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.4.1 Function Documentation

##### 17.4.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.5 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

## 17.6 args-parser.h File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/givinteger.h>
#include <givaro/givprint.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cstring>
#include <list>
#include <stdlib.h>
```

### Data Structures

- struct [Argument](#)

### Namespaces

- [FFLAS](#)

### Macros

- #define [TYPE\\_BOOL](#) [TYPE\\_NONE](#)
- #define [END\\_OF\\_ARGUMENTS](#) { '\0', "\0", "\0", [TYPE\\_NONE](#), NULL }
- #define [type\\_integer](#) long int

### Enumerations

- enum [ArgumentType](#) {  
[TYPE\\_NONE](#) , [TYPE\\_INT](#) , [TYPE\\_UINT64](#) , [TYPE\\_LONGLONG](#) ,  
[TYPE\\_INTEGER](#) , [TYPE\\_DOUBLE](#) , [TYPE\\_INTLIST](#) , [TYPE\\_STR](#) }

### Functions

- void [parseArguments](#) (int argc, char \*\*argv, [Argument](#) \*args, bool printDefaults=true)
- void [printHelpMessage](#) (const char \*program, [Argument](#) \*args, bool printDefaults=false)
- [Argument](#) \* [findArgument](#) ([Argument](#) \*args, char c)
- int [getListArgs](#) (std::list< int > &outlist, std::string &instring)  
*transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}*
- char \* [getArgumentValue](#) (int argc, char \*\*argv, int i)  
*Get the value of an argument and avoid core dump when no value was given after an argument.*
- std::ostream & [writeCommandString](#) (std::ostream &os, [Argument](#) \*args, const char \*programName=nullptr)  
*writes the values of all arguments, preceded by the programName*

#### 17.6.1 Macro Definition Documentation

### 17.6.1.1 TYPE\_BOOL

```
#define TYPE_BOOL TYPE_NONE
```

### 17.6.1.2 END\_OF\_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

### 17.6.1.3 type\_integer

```
#define type_integer long int
```

## 17.6.2 Enumeration Type Documentation

### 17.6.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

TYPE_NONE	
TYPE_INT	
TYPE_UINT64	
TYPE_LONGLONG	
TYPE_INTEGER	
TYPE_DOUBLE	
TYPE_INTLIST	
TYPE_STR	

## 17.6.3 Function Documentation

### 17.6.3.1 printHelpMessage()

```
void printHelpMessage (
    const char * program,
    Argument * args,
    bool printDefaults = false )
```

### 17.6.3.2 findArgument()

```
Argument* findArgument (
    Argument * args,
    char c )
```

### 17.6.3.3 getListArgs()

```
int getListArgs (
    std::list< int > & outlist,
    std::string & instring )
```

transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}

#### Parameters

<i>outlist</i>	list once converted
<i>instring</i>	list to be converted

#### Returns

status message.

## 17.7 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(m, n, r, t) (2.7*CUBE(double(n)/1000.0))/t`

### Typedefs

- `typedef Givaro::Timer TTimer`

### Functions

- `int main (int argc, char **argv)`

## 17.7.1 Macro Definition Documentation

### 17.7.1.1 CUBE

```
#define CUBE(
    x ) ( (x)*(x)*(x) )
```

### 17.7.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.7*CUBE(double(n)/1000.0))/t
```

## 17.7.2 Typedef Documentation

### 17.7.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.7.3 Function Documentation

### 17.7.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.8 benchmark-charpoly-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/Matio.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.8.1 Macro Definition Documentation

### 17.8.1.1 \_\_FFLASFFPACK\_FORCE\_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

## 17.8.2 Function Documentation

### 17.8.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.9 benchmark-charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<class Field >`  
`void run_with_field (int q, uint64_t bits, size_t n, size_t d, size_t iter, std::string file, int variant, uint64_t seed)`
- `int main (int argc, char **argv)`

### 17.9.1 Macro Definition Documentation

#### 17.9.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.9.2 Function Documentation

#### 17.9.2.1 run\_with\_field()

```
void run_with_field (
    int q,
    uint64_t bits,
    size_t n,
    size_t d,
    size_t iter,
    std::string file,
    int variant,
    uint64_t seed )
```

#### 17.9.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.10 benchmark-checkers.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

```
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include <fstream>
```

## Macros

- `#define ENABLE_ALL_CHECKINGS 1`
- `#define _NR_TESTS 5`
- `#define _MAX_SIZE_MATRICES 1000`
- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.10.1 Macro Definition Documentation

#### 17.10.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

#### 17.10.1.2 \_NR\_TESTS

```
#define _NR_TESTS 5
```

#### 17.10.1.3 \_MAX\_SIZE\_MATRICES

```
#define _MAX_SIZE_MATRICES 1000
```

#### 17.10.1.4 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

### 17.10.2 Function Documentation

#### 17.10.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.11 benchmark-dgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/config-blas.h"
```

```
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [CBLAS\\_GEMM](#) `cblas_dgemm`

## Typedefs

- typedef [FFLAS::Timer](#) `TTimer`
- typedef double [Floats](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.11.1 Macro Definition Documentation

#### 17.11.1.1 CBLAS\_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

### 17.11.2 Typedef Documentation

#### 17.11.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

#### 17.11.2.2 Floats

```
typedef double Floats
```

### 17.11.3 Function Documentation

#### 17.11.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.12 benchmark-dgetrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_HAVE_DGETRF 1`

## Typedefs

- `typedef FFLAS::Timer TTimer`

## Functions

- `int main (int argc, char **argv)`

## 17.12.1 Macro Definition Documentation

### 17.12.1.1 \_\_FFLASFFPACK\_HAVE\_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

## 17.12.2 Typedef Documentation

### 17.12.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.12.3 Function Documentation

### 17.12.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.13 benchmark-dgetri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- `typedef FFLAS::Timer TTimer`

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.13.1 Typedef Documentation

#### 17.13.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.13.2 Function Documentation

#### 17.13.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.14 benchmark-dsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [EFFGFF](#)(n, t, i) ( (double(n)/1000.\*double(n)/1000.\*double(n)/1000.0) / double(t) \* double(i) / 3.)

## Typedefs

- typedef [FFLAS::Timer](#) TTimer

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.14.1 Macro Definition Documentation

#### 17.14.1.1 EFFGFF

```
#define EFFGFF(
    n,
    t,
    i ) ( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i)
/ 3.)
```

## 17.14.2 Typedef Documentation

### 17.14.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.14.3 Function Documentation

### 17.14.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.15 benchmark-dtrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Typedefs

- typedef [FFLAS::Timer](#) TTimer

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.15.1 Typedef Documentation

### 17.15.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.15.2 Function Documentation

### 17.15.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.16 benchmark-dtrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_HAVE\_DTRTRI 1`

### Typedefs

- `typedef FFLAS::Timer TTimer`

### Functions

- `int main (int argc, char **argv)`

## 17.16.1 Macro Definition Documentation

### 17.16.1.1 `__FFLASFFPACK_HAVE_DTRTRI`

```
#define \_\_FFLASFFPACK\_HAVE\_DTRTRI 1
```

## 17.16.2 Typedef Documentation

### 17.16.2.1 `TTimer`

```
typedef FFLAS::Timer TTimer
```

## 17.16.3 Function Documentation

### 17.16.3.1 `main()`

```
int main (
    int argc,
    char ** argv )
```

## 17.17 benchmark-fadd-lvl2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.17.1 Macro Definition Documentation

#### 17.17.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.17.2 Function Documentation

#### 17.17.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.18 benchmark-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givrational.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<class Field > Field::Element run_with_field (int q, size_t iter, size_t N, const uint64_t BS, const size_t p, const size_t threads, uint64_t seed)`
- `int main (int argc, char **argv)`

### 17.18.1 Macro Definition Documentation

#### 17.18.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.18.2 Function Documentation

### 17.18.2.1 run\_with\_field()

```
Field::Element run_with_field (
    int q,
    size_t iter,
    size_t N,
    const uint64_t BS,
    const size_t p,
    const size_t threads,
    uint64_t seed )
```

### 17.18.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.19 benchmark-fgemm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `template<typename Ints >`  
`int tmain ()`
- `int main (int argc, char **argv)`

## 17.19.1 Macro Definition Documentation

### 17.19.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.19.2 Function Documentation

**17.19.2.1 tmain()**

```
int tmain ( )
```

**17.19.2.2 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.20 benchmark-fgemm-rns.C File Reference**

```
#include "fflas-ffpack/fflas/fflas.h"
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

**Typedefs**

- `typedef FFPACK::rns_double RNS`
- `typedef FFPACK::RNSInteger< RNS > Field`
- `typedef Field::Element_ptr Element_ptr`
- `typedef Field::ConstElement_ptr ConstElement_ptr`
- `typedef StrategyParameter::Threads THREADS`
- `typedef StrategyParameter::Grain GRAIN`
- `typedef StrategyParameter::TwoD TWOD`
- `typedef StrategyParameter::TwoDAdaptive TWODA`
- `typedef StrategyParameter::ThreeD THREED`
- `typedef StrategyParameter::ThreeDAdaptive THREEDA`
- `typedef StrategyParameter::ThreeDInPlace THREEDIP`
- `typedef ParSeqHelper::Sequential PSeq`

**Functions**

- `int main (int argc, char *argv[])`

**17.20.1 Macro Definition Documentation****17.20.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.20.2 Typedef Documentation****17.20.2.1 RNS**

```
typedef FFPACK::rns_double RNS
```

### 17.20.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

### 17.20.2.3 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

### 17.20.2.4 ConstElement\_ptr

```
typedef Field::ConstElement_ptr ConstElement_ptr
```

### 17.20.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

### 17.20.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

### 17.20.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

### 17.20.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

### 17.20.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

### 17.20.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

### 17.20.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

### 17.20.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

## 17.20.3 Function Documentation

### 17.20.3.1 main()

```
int main (
    int argc,
    char * argv[] )
```

## 17.21 benchmark-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CLASSIC\\_HYBRID](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.21.1 Macro Definition Documentation

#### 17.21.1.1 CLASSIC\_HYBRID

```
#define CLASSIC_HYBRID
```

### 17.21.2 Function Documentation

#### 17.21.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.22 benchmark-fgemv-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<typename T>`  
`std::ostream & write_matrix (std::ostream &out, Givaro::Integer p, size_t m, size_t n, T *C, size_t ldc)`

### 17.22.1 Macro Definition Documentation

#### 17.22.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.22.2 Function Documentation

#### 17.22.2.1 write\_matrix()

```
std::ostream& write_matrix (
    std::ostream & out,
    Givaro::Integer p,
    size_t m,
    size_t n,
    T * C,
    size_t ldc )
```

## 17.23 benchmark-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
```

## Data Structures

- `struct need_field_characteristic< Field >`
- `struct need_field_characteristic< Givaro::Modular< Field > >`
- `struct need_field_characteristic< Givaro::ModularBalanced< Field > >`
- `struct compatible_data_type< Field >`
- `struct compatible_data_type< Givaro::ZRing< float > >`
- `struct compatible_data_type< Givaro::ZRing< double > >`

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<class Field , class RandIter , class Matrix , class Vector >`  
`void fill_value (Field &F, RandIter &Rand, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK)`
- `template<class Field , class Matrix , class Vector >`  
`void genData (Field &F, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK, uint64_t bitsize, uint64_t seed)`
- `template<class Field , class Matrix , class Vector >`  
`bool check_result (Field &F, size_t m, size_t lda, Matrix &A, Vector &X, size_t incX, Vector &Y, size_t incY)`
- `template<class Field , class Matrix , class Vector >`  
`bool benchmark_with_timer (Field &F, int p, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, size_t iters, int t, double &time, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_disp (Field &F, bool pass, double &time, size_t iters, int p, size_t m, size_t k, arg &as)`
- `template<class Field , class arg >`  
`void benchmark_in_Field (Field &F, int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_with_field (int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_with_field (const Givaro::Integer &q, int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `int main (int argc, char **argv)`

## 17.23.1 Macro Definition Documentation

### 17.23.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.23.2 Function Documentation

### 17.23.2.1 fill\_value()

```
void fill_value (
    Field & F,
    RandIter & Rand,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
    size_t incY,
    size_t lda,
    int NBK )
```

### 17.23.2.2 genData()

```
void genData (
    Field & F,
    Matrix & A,
```

```
Vector & X,  
Vector & Y,  
size_t m,  
size_t k,  
size_t incX,  
size_t incY,  
size_t lda,  
int NBK,  
uint64_t bitsize,  
uint64_t seed )
```

### 17.23.2.3 check\_result()

```
bool check_result (   
    Field & F,  
    size_t m,  
    size_t lda,  
    Matrix & A,  
    Vector & X,  
    size_t incX,  
    Vector & Y,  
    size_t incY )
```

### 17.23.2.4 benchmark\_with\_timer()

```
bool benchmark_with_timer (   
    Field & F,  
    int p,  
    Matrix & A,  
    Vector & X,  
    Vector & Y,  
    size_t m,  
    size_t k,  
    size_t incX,  
    size_t incY,  
    size_t lda,  
    size_t iters,  
    int t,  
    double & time,  
    size_t GrainSize )
```

### 17.23.2.5 benchmark\_disp()

```
void benchmark_disp (   
    Field & F,  
    bool pass,  
    double & time,  
    size_t iters,  
    int p,  
    size_t m,  
    size_t k,  
    arg & as )
```

**17.23.2.6 benchmark\_in\_Field()**

```
void benchmark_in_Field (
    Field & F,
    int p,
    size_t m,
    size_t k,
    int NBK,
    uint64_t bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

**17.23.2.7 benchmark\_with\_field() [1/2]**

```
void benchmark_with_field (
    int p,
    size_t m,
    size_t k,
    int NBK,
    uint64_t bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

**17.23.2.8 benchmark\_with\_field() [2/2]**

```
void benchmark_with_field (
    const Givaro::Integer & q,
    int p,
    size_t m,
    size_t k,
    int NBK,
    uint64_t bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

**17.23.2.9 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.24 benchmark-fgesv.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.24.1 Macro Definition Documentation

#### 17.24.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.24.2 Function Documentation

#### 17.24.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.25 benchmark-fsyr2k.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Functions

- `int main (int argc, char **argv)`

### 17.25.1 Function Documentation

#### 17.25.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.26 benchmark-fsyrrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `int main (int argc, char **argv)`

### 17.26.1 Macro Definition Documentation

#### 17.26.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.26.2 Function Documentation

#### 17.26.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.27 benchmark-fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFPACK_FSYTRF_BC_CROUT`
- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

### Functions

- `int main (int argc, char **argv)`

## 17.27.1 Macro Definition Documentation

### 17.27.1.1 \_\_FFPACK\_FSYTRF\_BC\_CROUT

```
#define __FFPACK_FSYTRF_BC_CROUT
```

### 17.27.1.2 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.27.1.3 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.27.2 Function Documentation

### 17.27.2.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.28 benchmark-ftsrm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <vector>  
#include <string>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/utils/args-parser.h"  
#include "givaro/modular-integer.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET 1`

### Functions

- `int main (int argc, char **argv)`

## 17.28.1 Macro Definition Documentation

### 17.28.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.28.2 Function Documentation

### 17.28.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.29 benchmark-fftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `int main (int argc, char **argv)`

## 17.29.1 Macro Definition Documentation

### 17.29.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.29.2 Function Documentation

### 17.29.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.30 benchmark-fftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.30.1 Macro Definition Documentation

#### 17.30.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.30.2 Function Documentation

#### 17.30.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.31 benchmark-fftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.31.1 Macro Definition Documentation

#### 17.31.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

#### 17.31.1.2 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

## 17.31.2 Function Documentation

### 17.31.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.32 benchmark-inverse.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define CUBE(x) ((x)*(x)*(x))`

### Functions

- `int main (int argc, char **argv)`

## 17.32.1 Macro Definition Documentation

### 17.32.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

## 17.32.2 Function Documentation

### 17.32.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.33 benchmark-lqmp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "givaro/modular-integer.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.33.1 Function Documentation

#### 17.33.1.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.34 benchmark-lqup.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.34.1 Macro Definition Documentation

#### 17.34.1.1 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

### 17.34.2 Function Documentation

#### 17.34.2.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.35 benchmark-pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include <givaro/givranditer.h>
#include <iostream>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Macros

- #define `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET` 1
- #define `CUBE(x)` ((x)\*(x)\*(x))

### Typedefs

- typedef `Givaro::ModularBalanced< double > Field`

### Functions

- void `verification_PLUQ` (const `Field` &F, typename `Field::Element` \*B, typename `Field::Element` \*A, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)
- void `Rec_Initialize` (`Field` &F, `Field::Element` \*C, size\_t m, size\_t n, size\_t ldc)
- int `main` (int argc, char \*\*argv)

## 17.35.1 Macro Definition Documentation

### 17.35.1.1 `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET`

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.35.1.2 `CUBE`

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

## 17.35.2 Typedef Documentation

### 17.35.2.1 `Field`

```
typedef Givaro::ModularBalanced<double> Field
```

## 17.35.3 Function Documentation

**17.35.3.1 verification\_PLUQ()**

```
void verification_PLUQ (
    const Field & F,
    typename Field::Element * B,
    typename Field::Element * A,
    size_t * P,
    size_t * Q,
    size_t m,
    size_t n,
    size_t R )
```

**17.35.3.2 Rec\_Initialize()**

```
void Rec_Initialize (
    Field & F,
    Field::Element * C,
    size_t m,
    size_t n,
    size_t ldc )
```

**17.35.3.3 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.36 benchmark-quasisep.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- #define `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET` 1

**Functions**

- template<class Field >  
void `run_with_field` (int q, size\_t n, size\_t m, size\_t t, size\_t r, size\_t iter, uint64\_t seed)
- int `main` (int argc, char \*\*argv)

**17.36.1 Macro Definition Documentation**

### 17.36.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.36.2 Function Documentation

### 17.36.2.1 run\_with\_field()

```
void run_with_field (
    int q,
    size_t n,
    size_t m,
    size_t t,
    size_t r,
    size_t iter,
    uint64_t seed )
```

### 17.36.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.37 benchmark-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

## Data Structures

- class [Bench<Elt>](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.37.1 Function Documentation

### 17.37.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.38 benchmark-wino.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define `CUBE(x)`  $((x)*(x)*(x))$

### Functions

- template<class Field >  
void `launch_wino` (const `Field` &F, const size\_t &n, const size\_t &NB, const size\_t &wino, const bool &asmax, const size\_t &seed, const bool compare)
- int `main` (int argc, char \*\*argv)

## 17.38.1 Macro Definition Documentation

### 17.38.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

## 17.38.2 Function Documentation

### 17.38.2.1 launch\_wino()

```
void launch_wino (  
    const Field & F,  
    const size_t & n,  
    const size_t & NB,  
    const size_t & wino,  
    const bool & asmax,  
    const size_t & seed,  
    const bool compare )
```

### 17.38.2.2 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.39 bit\_manipulation.h File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

## Macros

- `#define __has_builtin(x) 0`

## Functions

- `int32_t clz (uint64_t val)`
- `int32_t clz (uint32_t val)`
- `int32_t ctz (uint32_t val)`
- `int32_t ctz (uint64_t val)`

### 17.39.1 Macro Definition Documentation

#### 17.39.1.1 \_\_has\_builtin

```
#define __has_builtin(  
    x ) 0
```

### 17.39.2 Function Documentation

#### 17.39.2.1 clz() [1/2]

```
int32_t clz (  
    uint64_t val ) [inline]
```

#### 17.39.2.2 clz() [2/2]

```
int32_t clz (  
    uint32_t val ) [inline]
```

#### 17.39.2.3 ctz() [1/2]

```
int32_t ctz (  
    uint32_t val ) [inline]
```

#### 17.39.2.4 ctz() [2/2]

```
int32_t ctz (  
    uint64_t val ) [inline]
```

## 17.40 blockcuts.inl File Reference

```
#include <fflas-ffpack/fflas/fflas_enum.h>  
#include <math.h>  
#include <cassert>
```

## Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)
- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)
- struct [Parallel< C, P >](#)
- struct [Sequential](#)
- struct [Compose< H1, H2 >](#)
- struct [ForStrategy1D< blocksize\\_t, Cut, Param >](#)
- struct [ForStrategy2D< blocksize\\_t, Cut, Param >](#)

## Namespaces

- [FFLAS](#)
- [FFLAS::CuttingStrategy](#)
- [FFLAS::StrategyParameter](#)
- [FFLAS::ParSeqHelper](#)

*ParSeqHelper* for both *fgemm* and *ftrsm*.

## Macros

- `#define __FFLASFFPACK_fflas_blockcuts_INL`
- `#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)`

## Typedefs

- typedef Row [RNSModulus](#)

## Functions

- `template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>  
void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`

- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`

## 17.40.1 Macro Definition Documentation

### 17.40.1.1 `__FFLASFFPACK_fflas_blockcuts_INL`

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

### 17.40.1.2 `__FFLASFFPACK_MINBLOCKCUTS`

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

## 17.41 `cast.h` File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- `template<class T, class CT = const T>  
T fflas\_const\_cast (CT x)`

## 17.42 `cblas.C` File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_CONFIGURATION`
- `#define \_\_FFLASFFPACK\_HAVE\_CBLAS 1`

### Functions

- `int main ()`

## 17.42.1 Macro Definition Documentation

### 17.42.1.1 `__FFLASFFPACK_CONFIGURATION`

```
#define __FFLASFFPACK_CONFIGURATION
```

### 17.42.1.2 \_\_FFLASFFPACK\_HAVE\_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

## 17.42.2 Function Documentation

### 17.42.2.1 main()

```
int main (
    void )
```

## 17.43 charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(m, n, r, t) (2.7\*CUBE(double(n)/1000.0))/t

### Typedefs

- typedef Givaro::Timer [TTimer](#)

### Functions

- int [main](#) ()

## 17.43.1 Macro Definition Documentation

### 17.43.1.1 CUBE

```
#define CUBE(
    x ) ( (x) * (x) * (x) )
```

### 17.43.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.7*CUBE(double(n)/1000.0))/t
```

## 17.43.2 Typedef Documentation

### 17.43.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.43.3 Function Documentation

### 17.43.3.1 main()

```
int main (
    void )
```

## 17.44 charpoly.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the characteristic polynomial of a matrix over a defined finite field.*

### 17.44.1 Function Documentation

#### 17.44.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the characteristic polynomial of a matrix over a defined finite field.  
Outputs the characteristic polynomial.

## 17.45 checker\_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Data Structures

- class [CheckerImplem\\_charpoly](#)< Field, Polynomial >
- class [CheckerImplem\\_charpoly](#)< Givaro::ZRing< Givaro::Integer >, Polynomial >

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_charpoly\\_INL](#)

## 17.45.1 Macro Definition Documentation

### 17.45.1.1 \_\_FFLASFFPACK\_checker\_charpoly\_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

## 17.46 checker\_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Data Structures

- class [CheckerImplem\\_Det< Field >](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_det\\_INL](#)

## 17.46.1 Macro Definition Documentation

### 17.46.1.1 \_\_FFLASFFPACK\_checker\_det\_INL

```
#define __FFLASFFPACK_checker_det_INL
```

## 17.47 checker\_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Data Structures

- struct [Checker\\_Empty< Field >](#)

### Namespaces

- [FFLAS](#)

## 17.48 checker\_fgemm.inl File Reference

### Data Structures

- class [CheckerImplem\\_fgemm< Field >](#)

### Namespaces

- [FFLAS](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_checker\\_fgemm\\_INL](#)

### 17.48.1 Macro Definition Documentation

#### 17.48.1.1 \_\_FFLASFFPACK\_checker\_fgemm\_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

## 17.49 checker\_ftsm.inl File Reference

### Data Structures

- class [CheckerImplem\\_ftsm](#)< Field >

### Namespaces

- [FFLAS](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_checker\\_ftsm\\_INL](#)

### 17.49.1 Macro Definition Documentation

#### 17.49.1.1 \_\_FFLASFFPACK\_checker\_ftsm\_INL

```
#define __FFLASFFPACK_checker_ftsm_INL
```

## 17.50 checker\_invert.inl File Reference

### Data Structures

- class [CheckerImplem\\_invert](#)< Field >

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- [#define \\_\\_FFLASFFPACK\\_checker\\_invert\\_INL](#)

### 17.50.1 Macro Definition Documentation

#### 17.50.1.1 \_\_FFLASFFPACK\_checker\_invert\_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

## 17.51 checker\_pluq.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Data Structures

- class [CheckerImplem\\_PLUQ< Field >](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_pluq\\_INL](#)

#### 17.51.1 Macro Definition Documentation

##### 17.51.1.1 \_\_FFLASFFPACK\_checker\_pluq\_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

## 17.52 checkers.doxy File Reference

## 17.53 checkers\_fflas.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
```

### Data Structures

- class [FailureFgemmCheck](#)
- class [FailureTrsmCheck](#)

### Namespaces

- [FFLAS](#)

### Typedefs

- template<class Field >  
using [Checker\\_fgemm](#) = [FFLAS::Checker\\_Empty< Field >](#)
- template<class Field >  
using [Checker\\_ftsm](#) = [FFLAS::Checker\\_Empty< Field >](#)

## 17.54 checkers\_fflas.inl File Reference

```
#include "checker_fgemm.inl"
#include "checker_ftrsm.inl"
```

### Namespaces

- [FFLAS](#)

### Macros

- #define [FFLASFFPACK\\_checkers\\_fflas\\_inl\\_H](#)

### Typedefs

- template<class Field >  
using [ForceCheck\\_fgemm](#) = CheckerImplem\_fgemm< [Field](#) >
- template<class Field >  
using [ForceCheck\\_ftrsm](#) = CheckerImplem\_ftrsm< [Field](#) >

## 17.54.1 Macro Definition Documentation

### 17.54.1.1 FFLASFFPACK\_checkers\_fflas\_inl\_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

## 17.55 checkers\_ffpack.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Data Structures

- class [FailurePLUQCheck](#)
- class [FailureDetCheck](#)
- class [FailureInvertCheck](#)
- class [FailureCharpolyCheck](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Typedefs

- template<class Field >  
using [Checker\\_PLUQ](#) = FFLAS::Checker\_Empty< [Field](#) >
- template<class Field >  
using [Checker\\_Det](#) = FFLAS::Checker\_Empty< [Field](#) >
- template<class Field >  
using [Checker\\_invert](#) = FFLAS::Checker\_Empty< [Field](#) >
- template<class Field , class Polynomial >  
using [Checker\\_charpoly](#) = FFLAS::Checker\_Empty< [Field](#) >

## 17.56 checkers\_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define FFLASFFPACK_checkers_ffpack_inl_H`

### Typedefs

- `template<class Field >`  
using [ForceCheck\\_PLUQ](#) = CheckerImplem\_PLUQ< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_Det](#) = CheckerImplem\_Det< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_invert](#) = CheckerImplem\_invert< [Field](#) >
- `template<class Field , class Polynomial >`  
using [ForceCheck\\_charpoly](#) = CheckerImplem\_charpoly< [Field](#), Polynomial >

### 17.56.1 Macro Definition Documentation

#### 17.56.1.1 FFLASFFPACK\_checkers\_ffpack\_inl\_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

## 17.57 clapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`
- `#define __FFLASFFPACK_HAVE_CLAPACK 1`

### Functions

- `int main ()`

### 17.57.1 Macro Definition Documentation

### 17.57.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

### 17.57.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.57.1.3 \_\_FFLASFFPACK\_HAVE\_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

## 17.57.2 Function Documentation

### 17.57.2.1 main()

```
int main (
    void )
```

## 17.58 config-blas.h File Reference

### Macros

- #define [CBLAS\\_INT](#) int
- #define [CBLAS\\_ENUM\\_DEFINED\\_H](#)
- #define [CBLAS\\_EXTERNALS](#)
- #define [blas\\_enum](#) enum

### Enumerations

- enum [CBLAS\\_ORDER](#) { [CblasRowMajor](#) =101 , [CblasColMajor](#) =102 }
- enum [CBLAS\\_TRANSPOSE](#) { [CblasNoTrans](#) =111 , [CblasTrans](#) =112 , [CblasConjTrans](#) =113 , [AtlasConj](#) =114 }
- enum [CBLAS\\_UPLO](#) { [CblasUpper](#) =121 , [CblasLower](#) =122 }
- enum [CBLAS\\_DIAG](#) { [CblasNonUnit](#) =131 , [CblasUnit](#) =132 }
- enum [CBLAS\\_SIDE](#) { [CblasLeft](#) =141 , [CblasRight](#) =142 }

### Functions

- void [daxpy\\_](#) (const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [saxpy\\_](#) (const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- double [ddot\\_](#) (const int \*, const double \*, const int \*, const double \*, const int \*)
- float [sdot\\_](#) (const int \*, const float \*, const int \*, const float \*, const int \*)
- double [dasum\\_](#) (const int \*, const double \*, const int \*)
- int [idamax\\_](#) (const int \*, const double \*, const int \*)
- double [dnrm2\\_](#) (const int \*, const double \*, const int \*)
- void [dgemv\\_](#) (const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- void [sgemv\\_](#) (const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void [dger\\_](#) (const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, double \*, const int \*)
- void [sger\\_](#) (const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, float \*, const int \*)

- void [dcopy\\_](#) (const int \*, const double \*, const int \*, double \*, const int \*)
- void [scopy\\_](#) (const int \*, const float \*, const int \*, float \*, const int \*)
- void [dscal\\_](#) (const int \*, const double \*, double \*, const int \*)
- void [sscal\\_](#) (const int \*, const float \*, float \*, const int \*)
- void [dtrsm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [strsm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- void [dtrmm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [strmm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- void [sgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void [dgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- void [cblas\\_dsyrk](#) (const enum [CBLAS\\_ORDER](#) Order, const enum [CBLAS\\_UPLO](#) Uplo, const enum [CBLAS\\_TRANSPOSE](#) Trans, const int N, const int K, const double alpha, const double \*A, const int lda, const double beta, double \*C, const int ldc)

## 17.58.1 Macro Definition Documentation

### 17.58.1.1 CBLAS\_INT

```
#define CBLAS_INT int
```

### 17.58.1.2 CBLAS\_ENUM\_DEFINED\_H

```
#define CBLAS_ENUM_DEFINED_H
```

### 17.58.1.3 CBLAS\_EXTERNALS

```
#define CBLAS_EXTERNALS
```

### 17.58.1.4 blas\_enum

```
#define blas_enum enum
```

## 17.58.2 Enumeration Type Documentation

### 17.58.2.1 CBLAS\_ORDER

```
enum CBLAS\_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

### 17.58.2.2 CBLAS\_TRANSPOSE

enum [CBLAS\\_TRANSPOSE](#)

#### Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

### 17.58.2.3 CBLAS\_UPLO

enum [CBLAS\\_UPLO](#)

#### Enumerator

CblasUpper	
CblasLower	

### 17.58.2.4 CBLAS\_DIAG

enum [CBLAS\\_DIAG](#)

#### Enumerator

CblasNonUnit	
CblasUnit	

### 17.58.2.5 CBLAS\_SIDE

enum [CBLAS\\_SIDE](#)

#### Enumerator

CblasLeft	
CblasRight	

## 17.58.3 Function Documentation

### 17.58.3.1 daxpy\_()

```
void daxpy_ (
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

### 17.58.3.2 saxpy\_()

```
void saxpy_ (
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

### 17.58.3.3 ddot\_()

```
double ddot_ (
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    const int * )
```

### 17.58.3.4 sdot\_()

```
float sdot_ (
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    const int * )
```

### 17.58.3.5 dasum\_()

```
double dasum_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.58.3.6 idamax\_()

```
int idamax_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.58.3.7 dnorm2\_()

```
double dnorm2_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.58.3.8 dgemv\_()

```
void dgemv_ (
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

### 17.58.3.9 sgemv\_()

```
void sgemv_ (
    const char * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    float * ,
    const int * )
```

### 17.58.3.10 dger\_()

```
void dger_ (
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

### 17.58.3.11 sger\_()

```
void sger_ (
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

**17.58.3.12 dcopy\_()**

```
void dcopy_ (
    const int * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

**17.58.3.13 scopy\_()**

```
void scopy_ (
    const int * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

**17.58.3.14 dscal\_()**

```
void dscal_ (
    const int * ,
    const double * ,
    double * ,
    const int * )
```

**17.58.3.15 sscal\_()**

```
void sscal_ (
    const int * ,
    const float * ,
    float * ,
    const int * )
```

**17.58.3.16 dtrsm\_()**

```
void dtrsm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

**17.58.3.17 strsm\_()**

```
void strsm_ (
    const char * ,
    const char * ,
```

```
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

#### 17.58.3.18 dtrmm\_()

```
void dtrmm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

#### 17.58.3.19 strmm\_()

```
void strmm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

#### 17.58.3.20 sgemm\_()

```
void sgemm_ (  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    float * ,  
    const int * )
```

### 17.58.3.21 dgemm\_()

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

### 17.58.3.22 cblas\_dsyrk()

```
void cblas_dsyrk (
    const enum CBLAS_ORDER Order,
    const enum CBLAS_UPLO Uplo,
    const enum CBLAS_TRANSPOSE Trans,
    const int N,
    const int K,
    const double alpha,
    const double * A,
    const int lda,
    const double beta,
    double * C,
    const int ldc )
```

## 17.59 config.h File Reference

### Macros

- #define HAVE\_BLAS 1
- #define HAVE\_CBLAS 1
- #define HAVE\_CXX11 1
- #define HAVE\_DLFCN\_H 1
- #define HAVE\_FLOAT\_H 1
- #define HAVE\_INT128 1
- #define HAVE\_INTTYPES\_H 1
- #define HAVE\_LAPACK 1
- #define HAVE\_LIMITS\_H 1
- #define HAVE\_LITTLE\_ENDIAN 1
- #define HAVE\_MEMORY\_H 1
- #define HAVE\_PTHREAD\_H 1
- #define HAVE\_STDDEF\_H 1
- #define HAVE\_STDINT\_H 1
- #define HAVE\_STDLIB\_H 1
- #define HAVE\_STRINGS\_H 1
- #define HAVE\_STRING\_H 1
- #define HAVE\_SYS\_STAT\_H 1

- `#define HAVE_SYS_TIME_H 1`
- `#define HAVE_SYS_TYPES_H 1`
- `#define HAVE_UNISTD_H 1`
- `#define LT_OBJDIR ".libs/"`
- `#define OPENBLAS_NUM_THREADS 1`
- `#define PACKAGE "fflas-ffpack"`
- `#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"`
- `#define PACKAGE_NAME "FFLAS-FFPACK"`
- `#define PACKAGE_STRING "FFLAS-FFPACK 2.5.0"`
- `#define PACKAGE_TARNAME "fflas-ffpack"`
- `#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"`
- `#define PACKAGE_VERSION "2.5.0"`
- `#define SIZEOF_CHAR 1`
- `#define SIZEOF_INT 4`
- `#define SIZEOF_LONG 8`
- `#define SIZEOF_LONG_LONG 8`
- `#define SIZEOF_SHORT 2`
- `#define SIZEOF__INT64_T 8`
- `#define STDC_HEADERS 1`
- `#define USE_OPENMP 1`
- `#define VERSION "2.5.0"`

## 17.59.1 Macro Definition Documentation

### 17.59.1.1 HAVE\_BLAS

```
#define HAVE_BLAS 1
```

### 17.59.1.2 HAVE\_CBLAS

```
#define HAVE_CBLAS 1
```

### 17.59.1.3 HAVE\_CXX11

```
#define HAVE_CXX11 1
```

### 17.59.1.4 HAVE\_DLFCN\_H

```
#define HAVE_DLFCN_H 1
```

### 17.59.1.5 HAVE\_FLOAT\_H

```
#define HAVE_FLOAT_H 1
```

### 17.59.1.6 HAVE\_INT128

```
#define HAVE_INT128 1
```

**17.59.1.7 HAVE\_INTTYPES\_H**

```
#define HAVE_INTTYPES_H 1
```

**17.59.1.8 HAVE\_LAPACK**

```
#define HAVE_LAPACK 1
```

**17.59.1.9 HAVE\_LIMITS\_H**

```
#define HAVE_LIMITS_H 1
```

**17.59.1.10 HAVE\_LITTLE\_ENDIAN**

```
#define HAVE_LITTLE_ENDIAN 1
```

**17.59.1.11 HAVE\_MEMORY\_H**

```
#define HAVE_MEMORY_H 1
```

**17.59.1.12 HAVE\_PTHREAD\_H**

```
#define HAVE_PTHREAD_H 1
```

**17.59.1.13 HAVE\_STDDEF\_H**

```
#define HAVE_STDDEF_H 1
```

**17.59.1.14 HAVE\_STDINT\_H**

```
#define HAVE_STDINT_H 1
```

**17.59.1.15 HAVE\_STDLIB\_H**

```
#define HAVE_STDLIB_H 1
```

**17.59.1.16 HAVE\_STRINGS\_H**

```
#define HAVE_STRINGS_H 1
```

**17.59.1.17 HAVE\_STRING\_H**

```
#define HAVE_STRING_H 1
```

**17.59.1.18 HAVE\_SYS\_STAT\_H**

```
#define HAVE_SYS_STAT_H 1
```

**17.59.1.19 HAVE\_SYS\_TIME\_H**

```
#define HAVE_SYS_TIME_H 1
```

**17.59.1.20 HAVE\_SYS\_TYPES\_H**

```
#define HAVE_SYS_TYPES_H 1
```

**17.59.1.21 HAVE\_UNISTD\_H**

```
#define HAVE_UNISTD_H 1
```

**17.59.1.22 LT\_OBJDIR**

```
#define LT_OBJDIR ".libs/"
```

**17.59.1.23 OPENBLAS\_NUM\_THREADS**

```
#define OPENBLAS_NUM_THREADS 1
```

**17.59.1.24 PACKAGE**

```
#define PACKAGE "fflas-ffpack"
```

**17.59.1.25 PACKAGE\_BUGREPORT**

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.59.1.26 PACKAGE\_NAME**

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

**17.59.1.27 PACKAGE\_STRING**

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

**17.59.1.28 PACKAGE\_TARNAME**

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

**17.59.1.29 PACKAGE\_URL**

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.59.1.30 PACKAGE\_VERSION**

```
#define PACKAGE_VERSION "2.5.0"
```

**17.59.1.31    SIZEOF\_CHAR**

```
#define SIZEOF_CHAR 1
```

**17.59.1.32    SIZEOF\_INT**

```
#define SIZEOF_INT 4
```

**17.59.1.33    SIZEOF\_LONG**

```
#define SIZEOF_LONG 8
```

**17.59.1.34    SIZEOF\_LONG\_LONG**

```
#define SIZEOF_LONG_LONG 8
```

**17.59.1.35    SIZEOF\_SHORT**

```
#define SIZEOF_SHORT 2
```

**17.59.1.36    SIZEOF\_\_\_INT64\_T**

```
#define SIZEOF___INT64_T 8
```

**17.59.1.37    STDC\_HEADERS**

```
#define STDC_HEADERS 1
```

**17.59.1.38    USE\_OPENMP**

```
#define USE_OPENMP 1
```

**17.59.1.39    VERSION**

```
#define VERSION "2.5.0"
```

## 17.60    config.h File Reference

### Macros

- [#define \\_\\_FFLASFFPACK\\_HAVE\\_BLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CBLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CXX11 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INT128 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LAPACK 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_MEMORY\\_H 1](#)

- `#define __FFLASFFPACK_HAVE_PTHREAD_H 1`
- `#define __FFLASFFPACK_HAVE_STDDEF_H 1`
- `#define __FFLASFFPACK_HAVE_STDINT_H 1`
- `#define __FFLASFFPACK_HAVE_STDLIB_H 1`
- `#define __FFLASFFPACK_HAVE_STRINGS_H 1`
- `#define __FFLASFFPACK_HAVE_STRING_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_STAT_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_TIME_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1`
- `#define __FFLASFFPACK_HAVE_UNISTD_H 1`
- `#define __FFLASFFPACK_LT_OBJDIR ".libs/"`
- `#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1`
- `#define __FFLASFFPACK_PACKAGE "fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"`
- `#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"`
- `#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.5.0"`
- `#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"`
- `#define __FFLASFFPACK_SIZEOF_CHAR 1`
- `#define __FFLASFFPACK_SIZEOF_INT 4`
- `#define __FFLASFFPACK_SIZEOF_LONG 8`
- `#define __FFLASFFPACK_SIZEOF_LONG_LONG 8`
- `#define __FFLASFFPACK_SIZEOF_SHORT 2`
- `#define __FFLASFFPACK_SIZEOF__INT64_T 8`
- `#define __FFLASFFPACK_STDC_HEADERS 1`
- `#define __FFLASFFPACK_USE_OPENMP 1`
- `#define __FFLASFFPACK_VERSION "2.5.0"`

## 17.60.1 Macro Definition Documentation

### 17.60.1.1 `__FFLASFFPACK_HAVE_BLAS`

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

### 17.60.1.2 `__FFLASFFPACK_HAVE_CBLAS`

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

### 17.60.1.3 `__FFLASFFPACK_HAVE_CXX11`

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

### 17.60.1.4 `__FFLASFFPACK_HAVE_DLFCN_H`

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

### 17.60.1.5 `__FFLASFFPACK_HAVE_FLOAT_H`

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

**17.60.1.6 \_\_FFLASFFPACK\_HAVE\_INT128**

```
#define __FFLASFFPACK_HAVE_INT128 1
```

**17.60.1.7 \_\_FFLASFFPACK\_HAVE\_INTTYPES\_H**

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

**17.60.1.8 \_\_FFLASFFPACK\_HAVE\_LAPACK**

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

**17.60.1.9 \_\_FFLASFFPACK\_HAVE\_LIMITS\_H**

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

**17.60.1.10 \_\_FFLASFFPACK\_HAVE\_LITTLE\_ENDIAN**

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

**17.60.1.11 \_\_FFLASFFPACK\_HAVE\_MEMORY\_H**

```
#define __FFLASFFPACK_HAVE_MEMORY_H 1
```

**17.60.1.12 \_\_FFLASFFPACK\_HAVE\_PTHREAD\_H**

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

**17.60.1.13 \_\_FFLASFFPACK\_HAVE\_STDDEF\_H**

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

**17.60.1.14 \_\_FFLASFFPACK\_HAVE\_STDINT\_H**

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

**17.60.1.15 \_\_FFLASFFPACK\_HAVE\_STDLIB\_H**

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

**17.60.1.16 \_\_FFLASFFPACK\_HAVE\_STRINGS\_H**

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

**17.60.1.17 \_\_FFLASFFPACK\_HAVE\_STRING\_H**

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

**17.60.1.18 \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H**

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

**17.60.1.19 \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

**17.60.1.20 \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

**17.60.1.21 \_\_FFLASFFPACK\_HAVE\_UNISTD\_H**

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

**17.60.1.22 \_\_FFLASFFPACK\_LT\_OBJDIR**

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

**17.60.1.23 \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS**

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

**17.60.1.24 \_\_FFLASFFPACK\_PACKAGE**

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

**17.60.1.25 \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT**

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.60.1.26 \_\_FFLASFFPACK\_PACKAGE\_NAME**

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

**17.60.1.27 \_\_FFLASFFPACK\_PACKAGE\_STRING**

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

**17.60.1.28 \_\_FFLASFFPACK\_PACKAGE\_TARNAME**

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

**17.60.1.29 \_\_FFLASFFPACK\_PACKAGE\_URL**

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.60.1.30 \_\_FFLASFFPACK\_PACKAGE\_VERSION**

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"
```

**17.60.1.31 \_\_FFLASFFPACK\_SIZEOF\_CHAR**

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

**17.60.1.32 \_\_FFLASFFPACK\_SIZEOF\_INT**

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

**17.60.1.33 \_\_FFLASFFPACK\_SIZEOF\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG 8
```

**17.60.1.34 \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

**17.60.1.35 \_\_FFLASFFPACK\_SIZEOF\_SHORT**

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

**17.60.1.36 \_\_FFLASFFPACK\_SIZEOF\_\_INT64\_T**

```
#define __FFLASFFPACK_SIZEOF__INT64_T 8
```

**17.60.1.37 \_\_FFLASFFPACK\_STDC\_HEADERS**

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

**17.60.1.38 \_\_FFLASFFPACK\_USE\_OPENMP**

```
#define __FFLASFFPACK_USE_OPENMP 1
```

**17.60.1.39 \_\_FFLASFFPACK\_VERSION**

```
#define __FFLASFFPACK_VERSION "2.5.0"
```

## 17.61 coo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::COO>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::COO\\_ZO>](#)

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix_t::COO > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)`

## 17.62 coo\_spmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmm\_INL`

### Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm\_simd\_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm\_simd\_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm\_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm\_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy, FieldCategories::GenericTag)`

- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.62.1 Macro Definition Documentation

### 17.62.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.63 coo\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.63.1 Macro Definition Documentation

### 17.63.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.64 coo\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_coo_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.64.1 Macro Definition Documentation

### 17.64.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

## 17.65 csr.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
```

### Data Structures

- `struct Sparse< _Field, SparseMatrix_t::CSR >`
- `struct Sparse< _Field, SparseMatrix_t::CSR_ZO >`

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)`

## 17.66 csr\_hyb.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmmm.inl"
```

## Data Structures

- `struct Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >`

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.67 csr\_hyb\_pspmm.inl File Reference

## Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL`

## Functions

- `template<class Field >`  
`void pfpmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-name Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag)`

- `template<class Field >`  
`void pfpsmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfpsmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfpsmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void pfpsmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfpsmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`

## 17.67.1 Macro Definition Documentation

### 17.67.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

## 17.68 csr\_hyb\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_pspmv\\_INL](#)

### Functions

- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

## 17.68.1 Macro Definition Documentation

### 17.68.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

## 17.69 csr\_hyb\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::Generic←`  
`Tag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::←`  
`UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`

### 17.69.1 Macro Definition Documentation

#### 17.69.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.70 csr\_hyb\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`

## 17.70.1 Macro Definition Documentation

### 17.70.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.71 csr\_hyb\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)< [ValT](#), [IdxT](#) >

### Namespaces

- [FFLAS](#)
- [FFLAS::csr\\_hyb\\_details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_utils\\_INL](#)

### Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A)
- template<class [Field](#) , class [IndexT](#) >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A, const [IndexT](#) \*row, const [IndexT](#) \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)

## 17.71.1 Macro Definition Documentation

### 17.71.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

## 17.72 csr\_pspmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_pspmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [pfsppmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int [Idx](#), typename [Field::Element\\_ptr](#) y\_, int [Idy](#), FieldCategories::GenericTag)

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.72.1 Macro Definition Documentation

### 17.72.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

## 17.73 csr\_pspmv.inl File Reference

```
#include <thread>
```

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.73.1 Macro Definition Documentation

### 17.73.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

## 17.74 csr\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`

- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t ↵`  
`blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, Field↵`  
`Categories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`

## 17.74.1 Macro Definition Documentation

### 17.74.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.75 csr\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.75.1 Macro Definition Documentation

### 17.75.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.76 csr\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)`
- `template<class Field >`  
`std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class IndexT >`  
`void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT >`  
`void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.77 cuda.C File Reference

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

### Functions

- int [main](#) ()

### 17.77.1 Function Documentation

#### 17.77.1.1 main()

```
int main (
    void )
```

## 17.78 debug.h File Reference

Various utilities for debugging.

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <iostream>
#include <sstream>
#include <cmath>
#include <stdexcept>
```

### Data Structures

- class [Failure](#)  
*A precondition failed.*

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [FFLASFFPACK\\_check](#)(check)
- #define [FFLASFFPACK\\_abort](#)(msg)

### Functions

- Failure & [failure](#) ()
- template<class T >  
bool [isOdd](#) (const T &a)
- bool [isOdd](#) (const float &a)
- bool [isOdd](#) (const double &a)

### 17.78.1 Detailed Description

Various utilities for debugging.

**Todo** we should put vector printing elsewhere.

## 17.78.2 Macro Definition Documentation

### 17.78.2.1 FFLASFFPACK\_check

```
#define FFLASFFPACK_check(  
    check )
```

**Value:**

```
if (!(check)) {\n    FFPACK::failure() (__func__, __FILE__, __LINE__, #check); \n    throw std::runtime_error(#check); \n}
```

### 17.78.2.2 FFLASFFPACK\_abort

```
#define FFLASFFPACK_abort(  
    msg )
```

**Value:**

```
{\n    FFPACK::failure() (__func__, __FILE__, __LINE__, msg); \n    throw std::runtime_error(msg); \n}
```

## 17.79 det.C File Reference

```
#include <givaro/modular.h>\n#include <iostream>\n#include "fflas-ffpack/fflas-ffpack-config.h"\n#include "fflas-ffpack/fflas-ffpack.h"\n#include "fflas-ffpack/utils/fflas_io.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the determinant of a matrix over a defined finite field.*

### 17.79.1 Function Documentation

#### 17.79.1.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

This example computes the determinant of a matrix over a defined finite field.

Outputs the determinant.

## 17.80 ell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_utils.inl"\n#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"\n#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
```

## Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_ZO>](#)

## Namespaces

- [FFLAS](#)

## Functions

- template<class Field, class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)
- template<class Field, class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)
- template<class Field>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL > &A)
- template<class Field>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A)

## 17.81 ell\_pspmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_pspmm\\_INL](#)

### Functions

- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag)
- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::GenericTag)
- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag)
- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, const int64\_t kmax)
- template<class Field>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, const int64\_t kmax)
- template<class Field, class Func>  
void [pfspmm\\_zo](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, Func &&func)

- `template<class Field , class Func >`  
`void pfsppmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`

## 17.81.1 Macro Definition Documentation

### 17.81.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

## 17.82 ell\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfsppmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.82.1 Macro Definition Documentation

### 17.82.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

## 17.83 ell\_simd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_simd>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO>](#)

### Namespaces

- [FFLAS](#)

### Functions

- template<class Field, class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)
- template<class Field, class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)
- template<class Field>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A)
- template<class Field>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A)

## 17.84 ell\_simd\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_pspmv\\_INL](#)

### Functions

- template<class Field>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class Field>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class Field>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const uint64\_t kmax)
- template<class Field>  
void [pfspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class Field>  
void [pfspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)

- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.84.1 Macro Definition Documentation

### 17.84.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

## 17.85 ell\_simd\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, type-`  
`name Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.85.1 Macro Definition Documentation

### 17.85.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

## 17.86 ell\_simd\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const`  
`IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row,`  
`const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.86.1 Macro Definition Documentation

### 17.86.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

## 17.87 ell\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

### 17.87.1 Macro Definition Documentation

#### 17.87.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

## 17.88 ell\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)

- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_spmv\\_INL](#)

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.88.1 Macro Definition Documentation

### 17.88.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

## 17.89 ell\_utils.inl File Reference

```
#include <vector>
```

## Namespaces

- [FFLAS](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_utils\\_INL](#)

## Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.89.1 Macro Definition Documentation

### 17.89.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

## 17.90 fblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define __FFLASFFPACK_CONFIGURATION`

### Functions

- `void dgemm_ (const char *, const char *, const int *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)`
- `int main ()`

## 17.90.1 Macro Definition Documentation

### 17.90.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

## 17.90.2 Function Documentation

### 17.90.2.1 dgemm\_()

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

### 17.90.2.2 main()

```
int main (
    void )
```

## 17.91 fflas-101\_1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.91.1 Function Documentation

#### 17.91.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.92 fflas-101\_3.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.92.1 Function Documentation

#### 17.92.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.93 fflas-ffpack-config.h File Reference

Defaults for optimised values.

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/fflas-ffpack-thresholds.h"
#include "fflas-ffpack/fflas-ffpack-default-thresholds.h"
#include "givaro/givconfig.h"
```

### Macros

- #define [GCC\\_VERSION](#) (\_\_GNUC\_\_ \* 10000 + \_\_GNUC\_MINOR\_\_ \* 100 + \_\_GNUC\_PATCHLEVEL\_\_)

### 17.93.1 Detailed Description

Defaults for optimised values.

While `fflas-ffpack-optimize.h` is created by `configure` script, (either left blank or filled by optimiser), this file produces the defaults for the optimised values. If `fflas-ffpack-optimize.h` is not empty, then its values precedes the defaults here.

### 17.93.2 Macro Definition Documentation

#### 17.93.2.1 GCC\_VERSION

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

## 17.94 fflas-ffpack-default-thresholds.h File Reference

### Macros

- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_PLUQ\\_THRESHOLD](#) 256
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#) 16
- #define [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#) 30
- #define [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#) 32
- #define [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#) 64
- #define [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#) 3000

### 17.94.1 Macro Definition Documentation

#### 17.94.1.1 \_\_FFLASFFPACK\_WINOTHRESHOLD

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

#### 17.94.1.2 \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

**17.94.1.3 \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL**

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

**17.94.1.4 \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT**

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

**17.94.1.5 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD**

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

**17.94.1.6 \_\_FFLASFFPACK\_CHARPOLY\_LUKrylov\_ArithProg\_THRESHOLD**

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

**17.94.1.7 \_\_FFLASFFPACK\_CHARPOLY\_Danilevskii\_LUKrylov\_THRESHOLD**

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

**17.94.1.8 \_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD**

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

**17.94.1.9 \_\_FFLASFFPACK\_FTRTRI\_THRESHOLD**

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

**17.94.1.10 \_\_FFLASFFPACK\_FSYTRF\_THRESHOLD**

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

**17.94.1.11 \_\_FFLASFFPACK\_FSYRK\_THRESHOLD**

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

**17.95 fflas-ffpack-thresholds.h File Reference****17.96 fflas-ffpack.dox File Reference****17.97 fflas-ffpack.h File Reference**

Includes [FFLAS](#) and [FFPACK](#).

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

**17.97.1 Detailed Description**

Includes [FFLAS](#) and [FFPACK](#).

## 17.98 fflas.doxy File Reference

## 17.99 fflas.h File Reference

### Finite Field Linear Algebra Subroutines

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include <cmath>
#include <cstring>
#include <float.h>
#include <algorithm>
#include "fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas_level1.inl"
#include "fflas_level2.inl"
#include "fflas_level3.inl"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas_freduce.h"
#include "fflas_fadd.h"
#include "fflas_fscal.h"
#include "fflas_fassign.h"
#include "fflas_fgemm.inl"
#include "fflas_pfgemm.inl"
#include "fflas_fgemv.inl"
#include "fflas-ffpack/paladin/pfgemv.inl"
#include "fflas_freivalds.inl"
#include "fflas_fger.inl"
#include "fflas_fsyrk.inl"
#include "fflas_fsyrk_strassen.inl"
#include "fflas_fsyr2k.inl"
#include "fflas_ftrsm.inl"
#include "fflas_pftrsm.inl"
#include "fflas_ftrmm.inl"
#include "fflas_ftrsv.inl"
#include "fflas_faxpy.inl"
#include "fflas_fdot.inl"
#include "fflas-ffpack/field/rns.h"
#include "fflas_fscal_mp.inl"
#include "fflas_freduce_mp.inl"
#include "fflas-ffpack/fflas/fflas_fger_mp.inl"
#include "fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas_ftrsm_mp.inl"
#include "fflas_fgemv_mp.inl"
#include "fflas-ffpack/field/rns.inl"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas_sparse.h"
#include "fflas-ffpack/checkers/checkers_fflas.inl"
```

### Macros

- `#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD`
- `#define DOUBLE_TO_FLOAT_CROSSOVER 800`

*Thresholds determining which floating point representation to use, depending on the cardinality of the finite field.*

### 17.99.1 Detailed Description

Finite Field Linear Algebra Subroutines

Author

Clément Pernet.

### 17.99.2 Macro Definition Documentation

#### 17.99.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

#### 17.99.2.2 DOUBLE\_TO\_FLOAT\_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 800
```

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field. This is only used when the element representation is not a floating point type.

**Bug** to be benchmarked.

## 17.100 fflas\_101.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.100.1 Function Documentation

#### 17.100.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.101 fflas\_101\_lvl1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.101.1 Function Documentation

### 17.101.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.102 fflas\_bounds.inl File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/flimits.h"
#include <givaro/udl.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_bounds\\_INL](#)
- [#define FFLAS\\_INT\\_TYPE](#) uint64\_t

## Functions

- [template<class Field >](#)  
double [computeFactorClassic](#) (const [Field](#) &F)
- [template<>](#) double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- [template<>](#) double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- [template<class Field >](#)  
size\_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- [Givaro::Integer](#) [InfNorm](#) (const size\_t M, const size\_t N, const [Givaro::Integer](#) \*A, const size\_t lda)
- [template<class Field >](#)  
size\_t [TRSMBound](#) (const [Field](#) &)  
*TRSMBound.*
- [template<class Element >](#)  
size\_t [TRSMBound](#) (const [Givaro::Modular](#)< Element > &F)  
*Specialization for positive modular representation over float.*
- [template<class Element >](#)  
size\_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< Element > &F)  
*Specialization for balanced modular representation over double.*

## 17.102.1 Macro Definition Documentation

### 17.102.1.1 \_\_FFLASFFPACK\_fflas\_bounds\_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

## 17.102.1.2 FFLAS\_INT\_TYPE

```
#define FFLAS_INT_TYPE uint64_t
```

## 17.103 fflas\_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

## Macros

- #define [FFLAS\\_COMPILED](#)

## Enumerations

- enum [FFLAS\\_C\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 , [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }
- *Storage by row or col ?*
- enum [FFLAS\\_C\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 , [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
- *Is matrix transposed ?*
- enum [FFLAS\\_C\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 , [FflasUpper](#) = 121 , [FflasLower](#) = 122 }
- *Is triangular matrix's shape upper ?*
- enum [FFLAS\\_C\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 , [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
- *Is the triangular matrix implicitly unit diagonal ?*
- enum [FFLAS\\_C\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 , [FflasLeft](#) = 141 , [FflasRight](#) = 142 }
- *On what side ?*
- enum [FFLAS\\_C\\_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }
- *FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.*

## Functions

- void [freducein\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [freduce\\_1\\_modular\\_double](#) (const double F, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fnegin\\_1\\_modular\\_double](#) (const double F, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [fneg\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fzero\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool [fiszero\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool [fequal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fassign\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void [fscale\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)

- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fscalin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t incA, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t incA, double \*B, const size\_t incB, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, double \*B, const size\_t incB, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t incA, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*x, const size\_t incX, const double \*y, const size\_t incY, double \*A, const size\_t incA, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t incA, double \*X, int incX, bool positive)
- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t incA, double \*B, const size\_t incB, bool positive)

- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

## 17.103.1 Macro Definition Documentation

### 17.103.1.1 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

## 17.103.2 Enumeration Type Documentation

### 17.103.2.1 FFLAS\_C\_ORDER

enum [FFLAS\\_C\\_ORDER](#)

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

### 17.103.2.2 FFLAS\_C\_TRANSPOSE

enum [FFLAS\\_C\\_TRANSPOSE](#)

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

### 17.103.2.3 FFLAS\_C\_UPLO

enum [FFLAS\\_C\\_UPLO](#)

Is triangular matrix's shape upper ?

## Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )
FflasUpper	
FflasLower	

**17.103.2.4 FFLAS\_C\_DIAG**enum [FFLAS\\_C\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

## Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )
FflasNonUnit	
FflasUnit	

**17.103.2.5 FFLAS\_C\_SIDE**enum [FFLAS\\_C\\_SIDE](#)

On what side ?

## Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.
FflasLeft	
FflasRight	

**17.103.2.6 FFLAS\_C\_BASE**enum [FFLAS\\_C\\_BASE](#)FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.  
(deprecated, should not be used)

## Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precision BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

**17.103.3 Function Documentation**

**17.103.3.1 freducein\_1\_modular\_double()**

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.103.3.2 freduce\_1\_modular\_double()**

```
void freduce_1_modular_double (
    const double F,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.103.3.3 fnegin\_1\_modular\_double()**

```
void fnegin_1_modular_double (
    const double F,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.103.3.4 fneg\_1\_modular\_double()**

```
void fneg_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.103.3.5 fzero\_1\_modular\_double()**

```
void fzero_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.103.3.6 fiszero\_1\_modular\_double()**

```
bool fiszero_1_modular_double (
    const double p,
    const size_t n,
```

```
    const double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.103.3.7 fequal\_1\_modular\_double()

```
bool fequal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.103.3.8 fassign\_1\_modular\_double()

```
void fassign_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.103.3.9 fscaln\_1\_modular\_double()

```
void fscaln_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.103.3.10 fscal\_1\_modular\_double()

```
void fscal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.103.3.11 faxpy\_1\_modular\_double()

```
void faxpy_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,
```

```
const double * X,  
const size_t incX,  
double * Y,  
const size_t incY,  
bool positive )
```

#### 17.103.3.12 fdot\_1\_modular\_double()

```
double fdot_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.103.3.13 fswap\_1\_modular\_double()

```
void fswap_1_modular_double (  
    const double p,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.103.3.14 fadd\_1\_modular\_double()

```
void fadd_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.103.3.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

**17.103.3.16 faddin\_1\_modular\_double()**

```
void faddin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.103.3.17 fsubin\_1\_modular\_double()**

```
void fsubin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.103.3.18 fassign\_2\_modular\_double()**

```
void fassign_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldB,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.19 fzero\_2\_modular\_double()**

```
void fzero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.20 fequal\_2\_modular\_double()**

```
bool fequal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    bool positive )
```

**17.103.3.21 fiszero\_2\_modular\_double()**

```
bool fiszero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.22 fidentity\_2\_modular\_double()**

```
void fidentity_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    const double d,
    bool positive )
```

**17.103.3.23 freducein\_2\_modular\_double()**

```
void freducein_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.24 freduce\_2\_modular\_double()**

```
void freduce_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldB,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.25 fnegin\_2\_modular\_double()**

```
void fnegin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.26 fneg\_2\_modular\_double()**

```
void fneg_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldB,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.27 fscaln\_2\_modular\_double()**

```
void fscaln_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
    const size_t ldA,
    bool positive )
```

**17.103.3.28 fscal\_2\_modular\_double()**

```
void fscal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )
```

**17.103.3.29 faxpy\_2\_modular\_double()**

```
void faxpy_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t ldX,
    double * Y,
    const size_t ldY,
    bool positive )
```

**17.103.3.30 fmove\_2\_modular\_double()**

```
void fmove_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
```

```
double * A,  
const size_t ldA,  
double * B,  
const size_t ldB,  
bool positive )
```

#### 17.103.3.31 fadd\_2\_modular\_double()

```
void fadd_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.103.3.32 fsub\_2\_modular\_double()

```
void fsub_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.103.3.33 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.103.3.34 faddin\_2\_modular\_double()

```
void faddin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * C,
```

```

    const size_t ldC,
    bool positive )

```

#### 17.103.3.35 fgemv\_2\_modular\_double()

```

double* fgemv_2_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE TransA,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * X,
    const size_t incX,
    const double betaA,
    double * Y,
    const size_t incY,
    bool positive )

```

#### 17.103.3.36 fger\_2\_modular\_double()

```

void fger_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * x,
    const size_t incX,
    const double * y,
    const size_t incY,
    double * A,
    const size_t ldA,
    bool positive )

```

#### 17.103.3.37 ftrsv\_2\_modular\_double()

```

void ftrsv_2_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t n,
    const double * A,
    const size_t ldA,
    double * X,
    int incX,
    bool positive )

```

#### 17.103.3.38 ftrsm\_3\_modular\_double()

```

void ftrsm_3_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_UPLO Uplo,

```

```

const enum FFLAS_C_TRANSPOSE TransA,
const enum FFLAS_C_DIAG Diag,
const size_t m,
const size_t n,
const double alpha,
const double * A,
const size_t ldA,
double * B,
const size_t ldB,
bool positive )

```

### 17.103.3.39 ftrmm\_3\_modular\_double()

```

void ftrmm_3_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )

```

### 17.103.3.40 fgemm\_3\_modular\_double()

```

double* fgemm_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const enum FFLAS_C_TRANSPOSE tB,
    const size_t m,
    const size_t n,
    const size_t k,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    const double betA,
    double * C,
    const size_t ldC,
    bool positive )

```

### 17.103.3.41 fsquare\_3\_modular\_double()

```

double* fsquare_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,

```

```
const double betA,
double * C,
const size_t ldC,
bool positive )
```

## 17.104 fflas\_enum.h File Reference

```
#include <algorithm>
```

### Data Structures

- class [AreEqual< X, Y >](#)
- class [AreEqual< X, X >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Enumerations

- enum [FFLAS\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }
- Storage by row or col ?*
- enum [FFLAS\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
- Is matrix transposed ?*
- enum [FFLAS\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 , [FflasLeftTri](#) = 123 , [FflasRightTri](#) = 124 }
- Is triangular matrix's shape upper ?*
- enum [FFLAS\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
- Is the triangular matrix implicitly unit diagonal ?*
- enum [FFLAS\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 }
- On what side ?*
- enum [FFLAS\\_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }
- FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*

### Functions

- template<class T >  
const T & [min3](#) (const T &m, const T &n, const T &k)
- template<class T >  
const T & [max3](#) (const T &m, const T &n, const T &k)
- template<class T >  
const T & [min4](#) (const T &m, const T &n, const T &k, const T &l)
- template<class T >  
const T & [max4](#) (const T &m, const T &n, const T &k, const T &l)

## 17.105 fflas\_fadd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas_fadd.inl"
```

### Data Structures

- struct [support\\_simd\\_add< T >](#)

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadding for symmetric matrices*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*

- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*

## 17.106 fflas\_fadd.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fadd_INL`

### Functions

- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)`
- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)`
- `template<class Element >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class Element >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class Field , bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵ Categories::ModularTag)`
- `template<class Field , bool ADD>`  
`std::enable_if< !FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵ Categories::ModularTag)`
- `template<class Field , bool ADD>`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵ Categories::GenericTag)`

- `template<class Field, bool ADD>`  
`std::enable_if<!FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field←`  
`Categories::UnparametricTag)`
- `template<class Field, bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field←`  
`Categories::UnparametricTag)`

## 17.106.1 Macro Definition Documentation

### 17.106.1.1 \_\_FFLASFFPACK\_fadd\_INL

```
#define __FFLASFFPACK_fadd_INL
```

## 17.107 fflas\_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

## 17.108 fflas\_fassign.inl File Reference

```
#include <string.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/debug.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fassign_INL`

## Functions

- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename`  
`Field::Element_ptr X, const size_t incX)`  

$$fassign : x \leftarrow y.$$
- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY,`  
`float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const`  
`size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY,`  
`float *X, const size_t incX)`
- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY,`  
`double *X, const size_t incX)`

- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >  
void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`

*fassign : A ← B.*

## 17.108.1 Macro Definition Documentation

### 17.108.1.1 \_\_FFLASFFPACK\_fassign\_INL

```
#define __FFLASFFPACK_fassign_INL
```

## 17.109 fflas\_faxpy.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::vectorised::unswitch](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_faxpy_INL`

### Functions

- `template<class Field >  
std::enable_if< !FFLAS::support_simd_mod< typename Field::Element >::value && FFLAS::support_fast_mod< typename Field::Element >::value, void >::type axypyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, HelperMod< Field > &H)`
- `template<class Field >  
std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type axypyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY, HelperMod< Field > &H)`
- `template<class Field >  
std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type axypyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n)`
- `template<class Field >  
std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type axypyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY)`
- `template<class Field >  
std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >  
void faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`

- template<class Field >  
void [faxpy](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  
$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- template<> void [faxpy](#) (const Givaro::DoubleDomain &, const size\_t N, const Givaro::DoubleDomain::Element a, [Givaro::DoubleDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::DoubleDomain::Element\\_ptr](#) y, const size\_t incy)
- template<> void [faxpy](#) (const Givaro::FloatDomain &, const size\_t N, const Givaro::FloatDomain::Element a, [Givaro::FloatDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::FloatDomain::Element\\_ptr](#) y, const size\_t incy)
- template<class Field >  
void [faxpy](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
$$faxpy : y \leftarrow \alpha \cdot x + y.$$

## 17.109.1 Macro Definition Documentation

### 17.109.1.1 \_\_FFLASFFPACK\_faxpy\_INL

```
#define __FFLASFFPACK_faxpy_INL
```

## 17.110 fflas\_fdot.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

## Namespaces

- [FFLAS](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fdot\\_INL](#)

## Functions

- template<class Field >  
[Field::Element fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<class Field >  
[Field::Element fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DelayedTag &MT)
- template<> Givaro::DoubleDomain::Element [fdot](#) (const Givaro::DoubleDomain &, const size\_t N, [Givaro::DoubleDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::DoubleDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<> Givaro::FloatDomain::Element [fdot](#) (const Givaro::FloatDomain &, const size\_t N, [Givaro::FloatDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::FloatDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<class Field, class T >  
[Field::Element fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::ConvertTo< T > &MT)
- template<class Field >  
[Field::Element fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultBoundedTag &dbt)

- `template<class Field >`  
`Field::Element fdot` (const `Field` &F, const `size_t` N, typename `Field::ConstElement_ptr` x, const `size_t` incx, typename `Field::ConstElement_ptr` y, const `size_t` incy, const `ParSeqHelper::Sequential` seq)
- `template<class Field >`  
`Field::Element fdot` (const `Field` &F, const `size_t` N, typename `Field::ConstElement_ptr` X, const `size_t` incX, typename `Field::ConstElement_ptr` Y, const `size_t` incY)  
 $fdot: \text{dot product } x^T y.$

## 17.110.1 Macro Definition Documentation

### 17.110.1.1 \_\_FFLASFFPACK\_fdot\_INL

```
#define __FFLASFFPACK_fdot_INL
```

## 17.111 fflas\_fgemm.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/debug.h"
#include "fflas_fgemm/fgemm_classical.inl"
#include "fflas_fgemm/fgemm_winograd.inl"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_fgemm\\_INL](#)

## Functions

- `template<class NewField , class Field , class FieldMode >`  
`Field::Element_ptr fgemm_convert` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Winograd`, `FieldMode` > &H)
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreAddReduction` (`Element` &Outmin, `Element` &Outmax, `Element` &Op1min, `Element` &Op1max, `Element` &Op2min, `Element` &Op2max, `MMHelper`< `Field`, `AlgoT`, `ModeCategories::LazyTag`, `ParSeqTrait` > &WH)
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreAddReduction` (`Element` &Outmin, `Element` &Outmax, `Element` &Op1min, `Element` &Op1max, `Element` &Op2min, `Element` &Op2max, `MMHelper`< `Field`, `AlgoT`, `ModeT`, `ParSeqTrait` > &WH)
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreSubReduction` (`Element` &Outmin, `Element` &Outmax, `Element` &Op1min, `Element` &Op1max, `Element` &Op2min, `Element` &Op2max, `MMHelper`< `Field`, `AlgoT`, `ModeCategories::LazyTag`, `ParSeqTrait` > &WH)
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreSubReduction` (`Element` &Outmin, `Element` &Outmax, `Element` &Op1min, `Element` &Op1max, `Element` &Op2min, `Element` &Op2max, `MMHelper`< `Field`, `AlgoT`, `ModeT`, `ParSeqTrait` > &WH)

- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)`
- `template<typename Field , class Cut , class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fgemm: Field GENERAL Matrix Multiply.*
- `template<typename Field , class ModeT , class ParSeq >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

*fsquare: Squares a matrix.*

- `template<class Field >`  
`Field::Element_ptr fsquareCommon` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, const `typename Field::Element` beta, `typename Field::Element_ptr` C, const `size_t` ldc)
- `template<> double * fsquare` (const `Givaro::ModularBalanced`< `double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `double` alpha, const `double` \*A, const `size_t` lda, const `double` beta, `double` \*C, const `size_t` ldc)
- `template<> float * fsquare` (const `Givaro::ModularBalanced`< `float` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `float` alpha, const `float` \*A, const `size_t` lda, const `float` beta, `float` \*C, const `size_t` ldc)
- `template<> double * fsquare` (const `Givaro::Modular`< `double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `double` alpha, const `double` \*A, const `size_t` lda, const `double` beta, `double` \*C, const `size_t` ldc)
- `template<> float * fsquare` (const `Givaro::Modular`< `float` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `float` alpha, const `float` \*A, const `size_t` lda, const `float` beta, `float` \*C, const `size_t` ldc)

### 17.111.1 Macro Definition Documentation

#### 17.111.1.1 \_\_FFLASFFPACK\_fgemm\_INL

```
#define __FFLASFFPACK_fgemm_INL
```

## 17.112 fflas\_fgemv.inl File Reference

```
#include <givaro/zring.h>
```

### Namespaces

- `FFLAS`
- `FFLAS::Protected`

### Macros

- `#define __FFLASFFPACK_fgemv_INL`

### Functions

- `template<typename FloatElement , class Field >`  
`Field::Element_ptr fgemv_convert` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, `typename Field::ConstElement_ptr` X, const `size_t` incX, const `typename Field::Element` beta, `typename Field::Element_ptr` Y, const `size_t` incY)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, `typename Field::ConstElement_ptr` X, const `size_t` incX, const `typename Field::Element` beta, `typename Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::MachineFloatTag` > > &H)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, `typename Field::ConstElement_ptr` X, const `size_t` incX, const `typename Field::Element` beta, `typename Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DelayedTag` > &H)

- template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::↵ DefaultTag > &H)
- template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::↵ LazyTag > &H)
- template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE TransA, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY)  
*finite prime Field GEneral Matrix Vector multiplication.*
- Givaro::ZRing< int64\_t >::Element\_ptr fgemv (const Givaro::ZRing< int64\_t > &F, const FFLAS\_↵ TRANSPOSE ta, const size\_t M, const size\_t N, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*X, const size\_t incX, const int64\_t beta, int64\_t \*Y, const size\_t incY, MMHelper< Givaro::↵ ZRing< int64\_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::DoubleDomain::Element\_ptr fgemv (const Givaro::DoubleDomain &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement\_ptr A, const size\_t lda, const Givaro::DoubleDomain::ConstElement\_ptr X, const size\_t incX, const Givaro::↵ DoubleDomain::Element beta, Givaro::DoubleDomain::Element\_ptr Y, const size\_t incY, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t ↵ N, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, type-↵ name Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::↵ ::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element\_ptr fgemv (const Givaro::FloatDomain &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement\_ptr A, const size\_t lda, const Givaro::FloatDomain::ConstElement\_ptr X, const size\_t incX, const Givaro::Float↵ Domain::Element beta, Givaro::FloatDomain::Element\_ptr Y, const size\_t incY, MMHelper< Givaro::Float↵ Domain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field, class Cut, class Param >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, ParSeqHelper::Parallel< Cut, Param > &parH)
- template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, ParSeqHelper::Sequential &seqH)

## 17.112.1 Macro Definition Documentation

### 17.112.1.1 \_\_FFLASFFPACK\_fgemv\_INL

```
#define __FFLASFFPACK_fgemv_INL
```

## 17.113 fflas\_fgmv\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fgmv_mp_INL`

### Functions

- [FFPACK::rns\\_double::Element\\_ptr fgmv](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- [FFPACK::rns\\_double::Element\\_ptr fgmv](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- [Givaro::Integer \\* fgmv](#) (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, MMHelper< [Givaro::ZRing](#)< [Givaro::Integer](#) >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< [ElementCategories::RNSElementTag](#) > > &H)
- [Givaro::Integer \\* fgmv](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, MMHelper< [Givaro::Modular](#)< [Givaro::Integer](#) >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< [ElementCategories::RNSElementTag](#) > > &H)
- `template<size_t K1, size_t K2, class ParSeq >`  
[RecInt::ruint](#)< K1 > \* [fgmv](#) (const [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const [RecInt::ruint](#)< K1 > alpha, const [RecInt::ruint](#)< K1 > \*A, const size\_t lda, const [RecInt::ruint](#)< K1 > \*X, const size\_t incx, [RecInt::ruint](#)< K1 > beta, [RecInt::ruint](#)< K1 > \*Y, const size\_t incy, MMHelper< [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< [ElementCategories::RNSElementTag](#) >, ParSeq > &H)

### 17.113.1 Macro Definition Documentation

#### 17.113.1.1 \_\_FFLASFFPACK\_fgmv\_mp\_INL

```
#define __FFLASFFPACK_fgmv_mp_INL
```

## 17.114 fflas\_fger.inl File Reference

### Namespaces

- [FFLAS](#)

- [FFLAS::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fger\\_INL](#)

## Functions

- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`  
*fger: rank one update of a general matrix*
- `template<class FloatElement , class Field >`  
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field , class AnyTag >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`
- `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)`

## 17.114.1 Macro Definition Documentation

### 17.114.1.1 \_\_FFLASFFPACK\_fger\_INL

```
#define __FFLASFFPACK_fger_INL
```

## 17.115 fflas\_fger\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFPACK_fger_mp_INL`

### Functions

- void [fger](#) (const Givaro::Modular< Givaro::Integer > &F, const size\_t M, const size\_t N, const typename Givaro::Integer alpha, typename Givaro::Integer \*x, const size\_t incx, typename Givaro::Integer \*y, const size\_t incy, typename Givaro::Integer \*A, const size\_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename RNS >  
void [fger](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSInteger](#)< [RNS](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename RNS >  
void [fger](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSIntegerMod](#)< [RNS](#) >, MMHelperAlgo::Classic > &H)

### 17.115.1 Macro Definition Documentation

#### 17.115.1.1 \_\_FFPACK\_fger\_mp\_INL

```
#define __FFPACK_fger_mp_INL
```

## 17.116 fflas\_freduce.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/field/field_traits.h"
#include "fflas-ffpack/utils/cast.h"
#include "fflas-ffpack/fflas/fflas_freduce.inl"
```

### Data Structures

- struct [support\\_simd\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< T >

- struct [support\\_fast\\_mod< float >](#)
- struct [support\\_fast\\_mod< double >](#)
- struct [support\\_fast\\_mod< int64\\_t >](#)

## Namespaces

- [FFLAS](#)

## Functions

- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow y \bmod F.$$
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow x \bmod F.$$
- template<class Field >  
void [freduce\\_constoverride](#) (const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) A, const size\_t incX)
- template<class Field, class ConstOtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t n, ConstOtherElement\_ptr Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)
- template<class Field >  
void [finit](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{finit Initializes } X \text{ in } F\$.$$
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{freduce } A \leftarrow A \bmod F.$$
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const FFLAS\_UPLO uplo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{freduce for square symmetric matrices}$$
- template<class Field >  
void [pfreduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t numths)
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{freduce } A \leftarrow B \bmod F.$$
- template<class Field >  
void [freduce\\_constoverride](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)
- template<class Field, class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{finit } A \leftarrow B \bmod F.$$
- template<class Field >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{finit Initializes } A \text{ in } F\$.$$

## 17.117 fflas\_freduce.inl File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

## Data Structures

- struct [HelperMod](#)< [Field](#), [ElementCategories::MachineIntTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::MachineFloatTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::ArbitraryPrecIntTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::FixedPrecIntTag](#) >

## Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::vectorised::unswitch](#)
- [FFLAS::details](#)

## Macros

- `#define \_\_FFLASFFPACK\_fflas\_freduce\_INL`
- `#define FFLASFFPACK\_COPY\_REDUCE 32 /* TODO TO BENCHMARK LATER */`

## Functions

- `template<class T >`  
`std::enable_if< ! std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<> Givaro::Integer reduce (Givaro::Integer A, Givaro::Integer B)`
- `float reduce (float A, float B, float invB, float min, float max)`
- `double reduce (double A, double B, double invB, double min, double max)`
- `int64_t reduce (int64_t A, int64_t p, double invp, double min, double max, int64_t pow50rem)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineIntTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineFloatTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > &H)`
- `template<class Field >`  
`std::enable_if<!FFLAS::support\_simd\_mod< typename Field::Element >::value &&FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, typename Field::Element\_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, const size_t &incX, typename Field::Element\_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, typename Field::Element\_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, const size_t &incX, typename Field::Element\_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::Element\_ptr A, const size_t incX, FieldCategories::ModularTag)`

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type` `freduce`  
`(const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename`  
`Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field, class FC >`  
`void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, type-`  
`name Field::Element_ptr A, const size_t incX, FC)`

### 17.117.1 Macro Definition Documentation

#### 17.117.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

#### 17.117.1.2 FFLASFFPACK\_COPY\_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCHMARK LATER */
```

## 17.118 fflas\_freduce\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_freduce\\_mp\\_INL](#)

### Functions

- `template<> void` `freduce` `(const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n,`  
`FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, size_t inc)`
- `template<> void` `freduce` `(const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m,`  
`const size_t n, FFPACK::rns_double::Element_ptr A, size_t lda)`

### 17.118.1 Macro Definition Documentation

#### 17.118.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

## 17.119 fflas\_freivalds.inl File Reference

### Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_freivalds_INL`

## Functions

- `template<class Field >`  
`bool freivalds (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m,`  
`const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A,`  
`const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::ConstElement_ptr`  
`C, const size_t ldc)`  
*freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

## 17.119.1 Macro Definition Documentation

### 17.119.1.1 \_\_FFLASFFPACK\_freivalds\_INL

```
#define __FFLASFFPACK_freivalds_INL
```

## 17.120 fflas\_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

## 17.121 fflas\_fscal.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::vectorised::unswitch](#)
- [FFLAS::details](#)

## Macros

- `#define __FFLASFFPACK_fscal_INL`

## Functions

- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typename Field::Element >::value &&FFLAS::support_fast_mod<`  
`typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field >`  
`&H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY, HelperMod< Field >`  
`&H)`

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscaln`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscal`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void fscaln`  
`(const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`  
`const size_t incX, FC)`
- `template<class Field, class FC >`  
`void fscal`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<class Field >`  
`void fscaln`  
`(const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr`  
`X, const size_t incX)`  

$$fscaln\ x \leftarrow \alpha \cdot x.$$
- `template<class Field >`  
`void fscal`  
`(const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$fscal\ y \leftarrow \alpha \cdot x.$$
- `template<> void fscal`  
`(const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr`  
`y, const size_t incy)`
- `template<> void fscal`  
`(const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const`  
`size_t incy)`
- `template<> void fscaln`  
`(const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscaln`  
`(const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void fscaln`  
`(const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::Element_ptr A, const size_t lda)`  

$$fscaln\ A \leftarrow a \cdot A.$$
- `template<class Field >`  
`void fscal`  
`(const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  

$$fscal\ B \leftarrow a \cdot A.$$

### 17.121.1 Macro Definition Documentation

### 17.121.1.1 \_\_FFLASFFPACK\_fscal\_INL

```
#define __FFLASFFPACK_fscal_INL
```

## 17.122 fflas\_fscal\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas_fscal.h"
#include "fflas_fgemm.inl"
#include "fflas-ffpack/fflas/fflas_freduce_mp.inl"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fscal_mp_INL`

### Functions

- `template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`

## 17.122.1 Macro Definition Documentation

### 17.122.1.1 \_\_FFLASFFPACK\_fscal\_mp\_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

## 17.123 fflas\_fsyr2k.inl File Reference

```
#include <fflas-ffpack/utils/fflas_io.h>
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_fsyrr2k_INL`

## Functions

- `template<class Field >`  
[Field::Element\\_ptr fsyrr2k](#) (const [Field](#) &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsyrr2k: Symmetric Rank 2K update*

## 17.123.1 Macro Definition Documentation

### 17.123.1.1 \_\_FFLASFFPACK\_fflas\_fsyrr2k\_INL

```
#define __FFLASFFPACK_fflas_fsyrr2k_INL
```

## 17.124 fflas\_fsyrr.inl File Reference

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- `#define __FFLASFFPACK_fflas_fsyrr_INL`

## Functions

- `template<class NewField , class Field , class FieldMode >`  
[Field::Element\\_ptr fsyrr\\_convert](#) (const [Field](#) &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Classic, FieldMode > &H)
- `template<class Field >`  
[Field::Element\\_ptr fsyrr](#) (const [Field](#) &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsyrr: Symmetric Rank K update*
- `template<class Field >`  
[Field::Element\\_ptr fsyrr](#) (const [Field](#) &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const Par↔SeqHelper::Sequential seq)
- `template<class Field >`  
[Field::Element\\_ptr fsyrr](#) (const [Field](#) &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)

- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::ConvertTo`< `ElementCategories::Machine`↵`FloatTag` >, `ParSeqHelper::Sequential` > &H)
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `size_t` N, const `typename` `Field::Element` alpha, `typename` `Field::Element_ptr` A, const `size_t` lda, const `MMHelper`< `Field`, `AlgoT`, `ModeCategories::LazyTag`, `ParSeqTrait` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DelayedTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::LazyTag` > &H)
- `template<class Field , typename Mode >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::DivideAndConquer`, `Mode` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultBoundedTag` > &H)
- `Givaro::FloatDomain::Element_ptr fsyrk` (const `Givaro::FloatDomain` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `Givaro::FloatDomain::Element` alpha, `Givaro::FloatDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::FloatDomain::Element` beta, `Givaro::FloatDomain::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Givaro::FloatDomain`, `MMHelperAlgo::`↵`Classic`, `ModeCategories::DefaultTag` > &H)
- `Givaro::DoubleDomain::Element_ptr fsyrk` (const `Givaro::DoubleDomain` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `Givaro::DoubleDomain::Element` alpha, `Givaro::DoubleDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::DoubleDomain::`↵`Element` beta, `Givaro::DoubleDomain::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Givaro::DoubleDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, `typename` `Field::Element_ptr` A, const `size_t` lda, `typename` `Field::ConstElement_ptr` D, const `size_t` incD, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, const `size_t` threshold=`__FFLASFFPACK_FSYRK_THRESHOLD`)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::Element_ptr` A, const `size_t` lda, `typename` `Field::ConstElement_ptr` D, const `size_t` incD, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq, const `size_t` threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::Element_ptr` A, const `size_t` lda, `typename` `Field::ConstElement_ptr` D, const `size_t` incD, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Parallel`< `Cut`, `Param` > par, const `size_t` threshold)

- template<class Field >  
Field::Element\_ptr fsyk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const std::vector< bool > &two← Block, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYK\_THRESHOLD)

*fsyk: Symmetric Rank K update with diagonal scaling*

## 17.124.1 Macro Definition Documentation

### 17.124.1.1 \_\_FFLASFFPACK\_fflas\_fsyk\_INL

```
#define __FFLASFFPACK_fflas_fsyk_INL
```

## 17.125 fflas\_fsyk\_strassen.inl File Reference

```
#include <givaro/givintsqrootmod.h>
```

### Namespaces

- FFLAS
- FFLAS::Protected

### Macros

- #define \_\_FFLASFFPACK\_fflas\_fsyk\_strassen\_INL

### Functions

- template<class Field , class Element , class AlgoT , class ParSeqTrait >  
bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >  
bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field , class Element , class AlgoT , class ParSeqTrait >  
bool NeedPreAxyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::← LazyTag, ParSeqTrait > &WH)
- template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >  
bool NeedPreAxyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field , class FieldTrait >  
void computeS1S2 (const Field &F, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element x, const typename Field::Element y, typename Field::Element\_ptr A, const size\_t← \_t lda, typename Field::Element\_ptr S, const size\_t lds, typename Field::Element\_ptr T, const size\_t ldt, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field >  
Field::Element\_ptr fsyk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)

- `template<class Field , class Mode >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, MMHelper<`Field`, MMHelperAlgo::Winograd, Mode > &H)
- `template<class Field , class FieldTrait >`  
`Field::Element_ptr fsyrk_strassen` (const `Field` &F, const FFLAS\_UPLO uplo, const FFLAS\_TRANSPOSE trans, const `size_t` N, const `size_t` K, const `typename` `Field::Element` y1, const `typename` `Field::Element` y2, const `typename` `Field::Element` alpha, `typename` `Field::Element_ptr` A, const `size_t` lda, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, MMHelper<`Field`, MMHelperAlgo::↵Winograd, FieldTrait > &WH)

## 17.125.1 Macro Definition Documentation

### 17.125.1.1 \_\_FFLASFFPACK\_fflas\_fsyrk\_strassen\_INL

```
#define __FFLASFFPACK_fflas_fsyrk_strassen_INL
```

## 17.126 fflas\_ftrmm.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_ftrmm\\_INL](#)

### Functions

- `template<class Field >`  
`void ftrmm` (const `Field` &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const `size_t` M, const `size_t` N, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, `typename` `Field::Element_ptr` B, const `size_t` ldb)  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- `template<class Field >`  
`void ftrmm` (const `Field` &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const `size_t` M, const `size_t` N, const `typename` `Field::Element` alpha, `typename` `Field::ConstElement_ptr` A, const `size_t` lda, `typename` `Field::ConstElement_ptr` B, const `size_t` ldb, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc)  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*

## 17.126.1 Macro Definition Documentation

### 17.126.1.1 \_\_FFLASFFPACK\_ftrmm\_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

## 17.127 fflas\_ftrsm.inl File Reference

### Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_ftrsm_INL`

## Functions

- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)`
- `template<class Field, class Cut, class Param >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)`
- `template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)`

### 17.127.1 Macro Definition Documentation

#### 17.127.1.1 \_\_FFLASFFPACK\_ftrsm\_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

## 17.128 fflas\_ftrsm\_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

```
#include <cmath>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFPACK_ftrsm_mp_INL`

## Functions

- void [ftrsm](#) (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*B, const size\_t ldb)
- void [cblas\\_impftrsm](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_SIDE Side, const enum FFLAS\_UPLO Uplo, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_DIAG Diag, const int M, const int N, const [FFPACK::rns\\_double\\_elt](#) alpha, [FFPACK::rns\\_double\\_elt\\_cstptr](#) A, const int lda, [FFPACK::rns\\_double\\_elt\\_ptr](#) B, const int ldb)

### 17.128.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

### 17.128.2 Macro Definition Documentation

#### 17.128.2.1 [\\_\\_FFPACK\\_ftrsm\\_mp\\_INL](#)

```
#define __FFPACK_ftrsm_mp_INL
```

## 17.129 fflas\_ftrsv.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrsv\\_INL](#)

### Functions

- template<class Field >  
void [ftrsv](#) (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)

*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

### 17.129.1 Macro Definition Documentation

#### 17.129.1.1 [\\_\\_FFLASFFPACK\\_ftrsv\\_INL](#)

```
#define __FFLASFFPACK_ftrsv_INL
```

## 17.130 fflas\_helpers.inl File Reference

```
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/flimits.h"
#include <algorithm>
```

## Data Structures

- struct [Auto](#)
- struct [Classic](#)
- struct [DivideAndConquer](#)
- struct [Winograd](#)
- struct [WinogradPar](#)
- struct [Bini](#)
- struct [AlgoChooser](#)< ModeT, ParSeq >
- struct [AlgoChooser](#)< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >
- struct [MMHelper](#)< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >

*FGEMM Helper for Default and ConvertTo modes of operation.*

- struct [MMHelper](#)< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >
- struct [MMHelper](#)< Field, AlgoTrait, ModeTrait, ParSeqTrait >
- struct [Recursive](#)
- struct [Iterative](#)
- struct [Hybrid](#)
- struct [TRSMHelper](#)< ReclterTrait, ParSeqTrait >

*TRSM Helper.*

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)
- [FFLAS::MMHelperAlgo](#)
- [FFLAS::StructureHelper](#)

*StructureHelper for ftrsm.*

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_fflas\\_mmhelper\\_INL](#)

## Functions

- [template<class Field >](#)  
[int WinogradSteps](#) (const [Field](#) &F, const [size\\_t](#) &m)  
*Computes the number of recursive levels to perform.*
- [template<class DFE >](#)  
[size\\_t min\\_types](#) (const DFE &k)
- [template<> size\\_t min\\_types](#) (const [Reclnt::rint](#)< 6 > &k)
- [template<> size\\_t min\\_types](#) (const [Reclnt::rint](#)< 7 > &k)
- [template<> size\\_t min\\_types](#) (const [Reclnt::rint](#)< 8 > &k)
- [template<> size\\_t min\\_types](#) (const [Reclnt::rint](#)< 9 > &k)
- [template<> size\\_t min\\_types](#) (const [Reclnt::rint](#)< 10 > &k)
- [template<> size\\_t min\\_types](#) (const [Givaro::Integer](#) &k)
- [template<class T >](#)  
[bool unfit](#) (T x)
- [template<> bool unfit](#) ([int64\\_t](#) x)
- [template<size\\_t K>](#)  
[bool unfit](#) ([Reclnt::rint](#)< K > x)
- [template<> bool unfit](#) ([Reclnt::rint](#)< 6 > x)

### 17.130.1 Macro Definition Documentation

### 17.130.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

## 17.131 fflas\_intrinsic.h File Reference

## 17.132 fflas\_io.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include <fstream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas_memory.h"
```

## Namespaces

- [FFLAS](#)

## Enumerations

- enum [FFLAS\\_FORMAT](#) {  
[FflasAuto](#) = 0 , [FflasDense](#) = 1 , [FflasSMS](#) = 2 , [FflasBinary](#) = 3 ,  
[FflasMath](#) = 4 , [FflasMaple](#) = 5 , [FflasSageMath](#) = 6 }

## Functions

- template<class Field >  
 std::ostream & [WriteMatrix](#) (std::ostream &c, const [Field](#) &F, size\_t m, size\_t n, typename [Field::ConstElement\\_ptr](#) A, size\_t lda, FFLAS\_FORMAT format, bool column\_major)  
*WriteMatrix: write a matrix to an output stream.*
- void [preamble](#) (std::ifstream &ifs, FFLAS\_FORMAT &format)
- template<class Field >  
[Field::Element\\_ptr](#) [ReadMatrix](#) (std::ifstream &ifs, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, FFLAS\_FORMAT format=FflasAuto)  
*ReadMatrix: read a matrix from an input stream.*
- template<class Field >  
[Field::Element\\_ptr](#) [ReadMatrix](#) (const std::string &matrix\_file, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, FFLAS\_FORMAT format=FflasAuto)  
*ReadMatrix: read a matrix from a file.*
- template<class Field >  
 void [WriteMatrix](#) (std::string &matrix\_file, const [Field](#) &F, int m, int n, typename [Field::ConstElement\\_ptr](#) A, size\_t lda, FFLAS\_FORMAT format=FflasDense, bool column\_major=false)  
*WriteMatrix: write a matrix to a file.*
- std::ostream & [WritePermutation](#) (std::ostream &c, const size\_t \*P, size\_t N)  
*WritePermutation: write a permutation matrix to an output stream.*

## 17.133 fflas\_L1\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

```
#include "fflas_L1_inst_implem.inl"
```

## Macros

- `#define __FFLAS_L1_INST_C`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.133.1 Macro Definition Documentation

### 17.133.1.1 \_\_FFLAS\_L1\_INST\_C

```
#define __FFLAS_L1_INST_C
```

### 17.133.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.133.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.133.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.133.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.133.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.133.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.133.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

**17.133.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.133.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.134 fflas\_L1\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

**Macros**

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

**17.134.1 Macro Definition Documentation****17.134.1.1 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**17.134.1.2 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.134.1.3 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.134.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.134.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.134.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.134.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.134.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.134.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.135 fflas\_L1\_inst\_implem.inl File Reference****Namespaces**

- [FFLAS](#)

**Functions**

- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow x \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{finit } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [fconvert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX, const [FFLAS\\_ELT](#) \*Y, const size\_t incY)  

$$\text{fconvert } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [fnegin](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fnegin } x \leftarrow -x.$$
- template [INST\\_OR\\_DECL](#) void [fneg](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fneg } x \leftarrow -y.$$
- template [INST\\_OR\\_DECL](#) void [fzero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fzero} : A \leftarrow 0.$$
- template [INST\\_OR\\_DECL](#) bool [fiszero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fiszero} : \text{test } X = 0.$$
- template [INST\\_OR\\_DECL](#) bool [fequal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*X, const size\_t incX, const [FFLAS\\_ELT](#) \*Y, const size\_t incY)  

$$\text{fequal} : \text{test } X = Y.$$
- template [INST\\_OR\\_DECL](#) void [fassign](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t N, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)

- fassign* :  $x \leftarrow y$ .
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)
  - fscal*  $x \leftarrow \alpha \cdot x$ .
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
  - fscal*  $y \leftarrow \alpha \cdot x$ .
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
  - faxpy* :  $y \leftarrow \alpha \cdot x + y$ .
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)
  - faxpy* :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
  - fswap* :  $X \leftrightarrow Y$ .
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)

## 17.136 fflas\_L2\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L2_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t

### 17.136.1 Macro Definition Documentation

**17.136.1.1 \_\_FFLAS\_L2\_INST\_C**

```
#define __FFLAS_L2_INST_C
```

**17.136.1.2 INST\_OR\_DECL**

```
#define INST_OR_DECL
```

**17.136.1.3 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.136.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.136.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.136.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.136.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.136.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.136.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.136.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.137 fflas\_L2\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

## Macros

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.137.1 Macro Definition Documentation

### 17.137.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.137.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.137.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.137.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.137.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.137.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.137.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.137.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.137.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.138 fflas\_L2\_inst\_implem.inl File Reference

### Namespaces

- [FFLAS](#)

### Functions

- template [INST\\_OR\\_DECL](#) void [fassign](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fassign : A \leftarrow B.$
- template [INST\\_OR\\_DECL](#) void [fzero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template [INST\\_OR\\_DECL](#) bool [fequal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fequal : test A = B.$
- template [INST\\_OR\\_DECL](#) bool [fiszero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fiszero : test A = 0.$
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) &d)  
 $creates a diagonal matrix$
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $creates a diagonal matrix$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $freduce A \leftarrow A mod F.$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $freduce A \leftarrow B mod F.$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $finit A \leftarrow B mod F.$
- template [INST\\_OR\\_DECL](#) void [fnegin](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fnegin A \leftarrow -A.$
- template [INST\\_OR\\_DECL](#) void [fneg](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fneg A \leftarrow -B.$
- template [INST\\_OR\\_DECL](#) void [fscaln](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fscaln A \leftarrow a \cdot A.$
- template [INST\\_OR\\_DECL](#) void [fscal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fscal B \leftarrow a \cdot A.$
- template [INST\\_OR\\_DECL](#) void [faxpy](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*X, const size\_t ldx, [FFLAS\\_ELT](#) \*Y, const size\_t ldy)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template [INST\\_OR\\_DECL](#) void [fmove](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fmove : y \leftarrow \alpha \cdot x + \beta \cdot y.$

- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fadd : matrix addition.*
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fsub : matrix subtraction.*
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fsubin  $C = C - B$*
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fadd : matrix addition with scaling.*
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*faddin*
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)  
*finite prime `FFLAS_FIELD`<`FFLAS_ELT`> GEneral Matrix Vector multiplication.*
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)  
*fger: rank one update of a general matrix*
- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, int incX)  
*ftsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

## 17.139 fflas\_L3\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L3_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t

## 17.139.1 Macro Definition Documentation

### 17.139.1.1 \_\_FFLAS\_L3\_INST\_C

```
#define __FFLAS_L3_INST_C
```

### 17.139.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.139.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.139.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.139.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.139.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.139.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.139.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.139.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.139.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.140 fflas\_L3\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

```
#include "fflas_L3_inst_implem.inl"
```

## Macros

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.140.1 Macro Definition Documentation

### 17.140.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.140.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.140.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.140.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.140.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.140.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.140.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.140.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

## 17.140.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.141 fflas\_L3\_inst\_implem.inl File Reference

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLAS__TRSM_READONLY`

## Functions

- template [INST\\_OR\\_DECL](#) void [ftrsm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrsm: **TR**iangular **S**ystem solve with **M**atrix.*

- template [INST\\_OR\\_DECL](#) void [ftrmm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrmm: **TR**iangular **M**atrix **M**ultiply.*

- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc)

*fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*

- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Sequential seq)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > par)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > par)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fsquare](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc)

*fsquare: Squares a matrix.*

## 17.141.1 Macro Definition Documentation

## 17.141.1.1 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.142 fflas\_level1.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_fflas_level1_INL`

### Functions

- `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$freduce\ x \leftarrow x \bmod F.$$
- `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$freduce\ x \leftarrow y \bmod F.$$
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$finit\ x \leftarrow y \bmod F.$$
- `template<class Field >`  
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$finit\ \text{Initializes } X \text{ in } F\mathbb{Z}.$$
- `template<class Field , class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fconvert\ x \leftarrow y \bmod F.$$
- `template<class Field >`  
`void fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$fnegin\ x \leftarrow -x.$$
- `template<class Field >`  
`void fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$fneg\ x \leftarrow -y.$$
- `template<class Field >`  
`void fzero (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$fzero : A \leftarrow 0.$$
- `template<class Field , class Randlter >`  
`void frand (const Field &F, Randlter &G, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$frand : A \leftarrow random.$$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX)`  

$$fiszero : test\ X = 0.$$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fequal : test\ X = Y.$$
- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$fassign : x \leftarrow y.$$

- template<class Field >  
void **fscaln** (const Field &F, const size\_t n, const typename Field::Element alpha, typename Field::Element\_ptr X, const size\_t incX)  
 $fscaln\ x \leftarrow \alpha \cdot x.$
- template<class Field >  
void **fscal** (const Field &F, const size\_t n, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
 $fscal\ y \leftarrow \alpha \cdot x.$
- template<class Field >  
void **faxpy** (const Field &F, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template<class Field >  
void **faxpby** (const Field &F, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template<class Field >  
Field::Element **fdot** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)  
 $fdot: dot\ product\ x^T y.$
- template<class Field >  
Field::Element **fdot** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr x, const size\_t incx, typename Field::ConstElement\_ptr y, const size\_t incy, const ParSeqHelper::Sequential seq)
- template<typename Field, class Cut, class Param >  
Field::Element **fdot** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field >  
void **fswap** (const Field &F, const size\_t N, typename Field::Element\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
 $fswap: X \leftrightarrow Y.$
- template<class Field >  
void **pfadd** (const Field &F, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, typename Field::Element\_ptr C, const size\_t ldc, const size\_t numths)
- template<class Field >  
void **pfsub** (const Field &F, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, typename Field::Element\_ptr C, const size\_t ldc, const size\_t numths)
- template<class Field >  
void **pfaddin** (const Field &F, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr B, const size\_t ldb, typename Field::Element\_ptr C, const size\_t ldc, size\_t numths)
- template<class Field >  
void **pfsubin** (const Field &F, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr B, const size\_t ldb, typename Field::Element\_ptr C, const size\_t ldc, size\_t numths)
- template<class Field >  
void **fadd** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t inca, typename Field::ConstElement\_ptr B, const size\_t incb, typename Field::Element\_ptr C, const size\_t incc)
- template<class Field >  
void **fsb** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t inca, typename Field::ConstElement\_ptr B, const size\_t incb, typename Field::Element\_ptr C, const size\_t incc)
- template<class Field >  
void **faddin** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr B, const size\_t incb, typename Field::Element\_ptr C, const size\_t incc)
- template<class Field >  
void **fsubin** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr B, const size\_t incb, typename Field::Element\_ptr C, const size\_t incc)

- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename`  
`Field::Element_ptr C, const size_t incc)`

## 17.142.1 Macro Definition Documentation

### 17.142.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

## 17.143 fflas\_level2.inl File Reference

```
#include "givaro/zring.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_fflas_level2_INL`

## Functions

- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const`  
`size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
 $fassign : A \leftarrow B.$
- `template<class Field >`  
`void fzero (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
 $fzero : A \leftarrow 0.$
- `template<class Field >`  
`void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename`  
`Field::Element_ptr A, const size_t lda)`  
 $fzero : A \leftarrow 0 \text{ for a triangular matrix.}$
- `template<class Field, class Randlter >`  
`void frand (const Field &F, Randlter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const`  
`size_t lda)`  
 $frand : A \leftarrow \text{random.}$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr B, const size_t ldb)`  
 $fequal : \text{test } A = B.$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t`  
`lda)`  
 $fiszero : \text{test } A = 0.$
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda,`  
`const typename Field::Element &d)`  
 $\text{creates a diagonal matrix}$

- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*creates a diagonal matrix*
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*freduce  $A \leftarrow A \bmod F$ .*
- `template<class Field >`  
`void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*freduce for square symmetric matrices*
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*freduce  $A \leftarrow B \bmod F$ .*
- `template<class Field, class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*finit  $A \leftarrow B \bmod F$ .*
- `template<class Field, class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*finit Initializes  $A$  in  $F\mathbb{F}$ .*
- `template<class Field, class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  
*fconvert  $A \leftarrow B \bmod F$ .*
- `template<class Field >`  
`void fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*fnegin  $A \leftarrow -A$ .*
- `template<class Field >`  
`void fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*fneg  $A \leftarrow -B$ .*
- `template<class Field >`  
`void fscaln (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)`  
*fscaln  $A \leftarrow a \cdot A$ .*
- `template<class Field >`  
`void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*fscal  $B \leftarrow a \cdot A$ .*
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`  
*faxpy :  $y \leftarrow \alpha \cdot x + y$ .*
- `template<class Field >`  
`void faxpby (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t ldy)`  
*faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .*
- `template<class Field >`  
`void fmove (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .*

- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadding for symmetric matrices*
- `template<class Field >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)`  
*finite prime Field GEneral Matrix Vector multiplication.*
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`  
*fger: rank one update of a general matrix*
- `template<class Field >`  
`void ftrsv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)`  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*
- `template<class Field >`  
`size_t bitsize (const Field &F, size_t M, size_t N, const typename Field::ConstElement_ptr A, size_t lda)`  
*bitsize: Computes the largest bitsize of the matrix' coefficients.*
- `template<> size_t bitsize< Givaro::ZRing< Givaro::Integer > > (const Givaro::ZRing< Givaro::Integer > &F, size_t M, size_t N, const Givaro::Integer *A, size_t lda)`
- `template<class Field >`  
`void ftrmv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)`  
*ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$*

### 17.143.1 Macro Definition Documentation

## 17.143.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

## 17.144 fflas\_level3.inl File Reference

```
#include "fflas_bounds.inl"
#include "fflas_helpers.inl"
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- `#define __FFLASFFPACK_fflas_fflas_level3_INL`
- `#define __FFLAS__TRSM_READONLY`

## Functions

- `template<class Field >`  
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, type-`  
`name Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename`  
`Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrsm: **TR**angular **S**ystem solve with **M**atrix.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const`  
`size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const`  
`size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyrk: Symmetric Rank K update*
- `template<class Field, typename FieldTrait >`  
`Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE`  
`trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2,`  
`const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename`  
`Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::↵`  
`Winograd, FieldTrait > &H)`

- `template<class Field >`  
`Field::Element_ptr fsyr2k` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyr2k: Symmetric Rank 2K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Sequential seq, const size\_t threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const std::vector< bool > &two← Block, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<typename Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fgemm: Field GENeral Matrix Multiply.*
- `template<typename Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::← Sequential seq)
- `template<typename Field , class Cut , class Param >`  
`Field::Element_ptr fgemm` (const `Field` &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::← Parallel< Cut, Param > par)
- `template<typename Field >`  
`Field::Element_ptr pfgemm` (const `Field` &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, size\_t numthreads=0)
- `template<class Field >`  
`Field::Element * pfgemm_1D_rec` (const `Field` &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_← TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha,

```
const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t
ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)
```

- template<class Field >  
Field::Element \* pfgemm\_2D\_rec (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha,  
const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t  
ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
Field::Element \* pfgemm\_3D\_rec (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha,  
const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t  
ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil,  
size\_t \*x)
- template<class Field >  
Field::Element\_ptr pfgemm\_3D\_rec2 (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const  
typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const  
typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil, size\_t \*x)
- template<class Field >  
Field::Element\_ptr fsquare (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename  
Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element  
beta, typename Field::Element\_ptr C, const size\_t ldc)

*fsquare: Squares a matrix.*

## 17.144.1 Macro Definition Documentation

### 17.144.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

### 17.144.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.145 fflas\_lvl1.C File Reference

C functions calls for level 1 FFLAS in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

## Functions

- void [freducein\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [freduce\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fnegin\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [fneg\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fzero\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool [fiszero\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)

- bool [fequal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fassign\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fscalin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)

### 17.145.1 Detailed Description

C functions calls for level 1 [FFLAS](#) in `fflas-c.h`.

Author

Brice Boyer

See also

[fflas/fflas\\_level1.inl](#)

### 17.145.2 Function Documentation

#### 17.145.2.1 `freducein_1_modular_double()`

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

#### 17.145.2.2 `freduce_1_modular_double()`

```
void freduce_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.145.2.3 fnegin\_1\_modular\_double()**

```
void fnegin_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.145.2.4 fneg\_1\_modular\_double()**

```
void fneg_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.145.2.5 fzero\_1\_modular\_double()**

```
void fzero_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.145.2.6 fiszero\_1\_modular\_double()**

```
bool fiszero_1_modular_double (
    const double p,
    const size_t n,
    const double * X,
    const size_t incX,
    bool positive )
```

**17.145.2.7 fequal\_1\_modular\_double()**

```
bool fequal_1_modular_double (
    const double p,
    const size_t n,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    bool positive )
```

**17.145.2.8 fassign\_1\_modular\_double()**

```
void fassign_1_modular_double (
    const double p,
    const size_t n,
```

```
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.145.2.9 fscalin\_1\_modular\_double()

```
void fscalin_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.145.2.10 fscal\_1\_modular\_double()

```
void fscal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.145.2.11 faxpy\_1\_modular\_double()

```
void faxpy_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.145.2.12 fdot\_1\_modular\_double()

```
double fdot_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.145.2.13 fswap\_1\_modular\_double()

```
void fswap_1_modular_double (  

```

```
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

#### 17.145.2.14 fadd\_1\_modular\_double()

```
void fadd_1_modular_double (
    const double p,
    const size_t n,
    const double * A,
    const size_t incA,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

#### 17.145.2.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (
    const double p,
    const size_t n,
    const double * A,
    const size_t incA,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

#### 17.145.2.16 faddin\_1\_modular\_double()

```
void faddin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

#### 17.145.2.17 fsubin\_1\_modular\_double()

```
void fsubin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

## 17.146 fflas\_lvl2.C File Reference

C functions calls for level 2 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

### Functions

- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fscalin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, bool positive)
- void [fscale\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, double \*A, const size\_t lda, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t lda, double \*X, int incX, bool positive)

## 17.146.1 Detailed Description

C functions calls for level 2 [FFLAS](#) in fflas-c.h.

### Author

Brice Boyer

### See also

[fflas/fflas\\_level2.inl](#)

## 17.146.2 Function Documentation

### 17.146.2.1 fassign\_2\_modular\_double()

```
void fassign_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

### 17.146.2.2 fzero\_2\_modular\_double()

```
void fzero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

### 17.146.2.3 fequal\_2\_modular\_double()

```
bool fequal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    const double * B,
    const size_t ldb,
    bool positive )
```

### 17.146.2.4 fiszero\_2\_modular\_double()

```
bool fiszero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
```

```
    const size_t lda,  
    bool positive )
```

#### 17.146.2.5 fidentity\_2\_modular\_double()

```
void fidentity_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    const double d,  
    bool positive )
```

#### 17.146.2.6 freducein\_2\_modular\_double()

```
void freducein_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.146.2.7 freduce\_2\_modular\_double()

```
void freduce_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

#### 17.146.2.8 fnegin\_2\_modular\_double()

```
void fnegin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.146.2.9 fneg\_2\_modular\_double()

```
void fneg_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,
```

```
double * B,  
const size_t ldb,  
bool positive )
```

#### 17.146.2.10 fscaln\_2\_modular\_double()

```
void fscaln_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.146.2.11 fscal\_2\_modular\_double()

```
void fscal_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

#### 17.146.2.12 faxpy\_2\_modular\_double()

```
void faxpy_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

#### 17.146.2.13 fmove\_2\_modular\_double()

```
void fmove_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

**17.146.2.14 fadd\_2\_modular\_double()**

```
void fadd_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.146.2.15 fsub\_2\_modular\_double()**

```
void fsub_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.146.2.16 fsubin\_2\_modular\_double()**

```
void fsubin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.146.2.17 faddin\_2\_modular\_double()**

```
void faddin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.146.2.18 fgemv\_2\_modular\_double()**

```
double* fgemv_2_modular_double (
    const double p,
```

```

const enum FFLAS_C_TRANSPOSE TransA,
const size_t m,
const size_t n,
const double alpha,
const double * A,
const size_t lda,
const double * X,
const size_t incX,
const double beta,
double * Y,
const size_t incY,
bool positive )

```

#### 17.146.2.19 fger\_2\_modular\_double()

```

void fger_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    double * A,
    const size_t lda,
    bool positive )

```

#### 17.146.2.20 ftrsv\_2\_modular\_double()

```

void ftrsv_2_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t n,
    const double * A,
    const size_t lda,
    double * X,
    int incX,
    bool positive )

```

## 17.147 fflas\_lvl3.C File Reference

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

```

#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"

```

### Functions

- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const

size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)

- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

### 17.147.1 Detailed Description

C functions calls for level 3 [FFLAS](#) in [fflas-c.h](#).

Author

Brice Boyer

See also

[fflas/fflas\\_level3.inl](#)

### 17.147.2 Function Documentation

#### 17.147.2.1 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (
    const double p,
    const enum FFLAS\_C\_SIDE Side,
    const enum FFLAS\_C\_UPLO Uplo,
    const enum FFLAS\_C\_TRANSPOSE tA,
    const enum FFLAS\_C\_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )
```

#### 17.147.2.2 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (
    const double p,
    const enum FFLAS\_C\_SIDE Side,
    const enum FFLAS\_C\_UPLO Uplo,
    const enum FFLAS\_C\_TRANSPOSE tA,
    const enum FFLAS\_C\_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
```

```

    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )

```

### 17.147.2.3 fgemm\_3\_modular\_double()

```

double* fgemm_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const enum FFLAS_C_TRANSPOSE tB,
    const size_t m,
    const size_t n,
    const size_t k,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    const double betaA,
    double * C,
    const size_t ldC,
    bool positive )

```

### 17.147.2.4 fsquare\_3\_modular\_double()

```

double* fsquare_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double betaA,
    double * C,
    const size_t ldC,
    bool positive )

```

## 17.148 fflas\_memory.h File Reference

```

#include "fflas-ffpack/utils/align-allocator.h"
#include <givaro/givinteger.h>

```

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Element >`  
`bool alignable ()`
- `template<> bool alignable< Givaro::Integer * > ()`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)`

- `template<class Field >`  
`Field::Element_ptr fflas_new` (const `Field` &F, const `size_t` m, const `size_t` n, const `Alignment` align=`Alignment::DEFAULT`)
- `template<class Element >`  
`Element * fflas_new` (const `size_t` m, const `Alignment` align=`Alignment::DEFAULT`)
- `template<> void fflas_delete` (`FFPACK::rns_double_elt_ptr` A)
- `template<class Ptr, class ... Args>`  
`void fflas_delete` (`Ptr` p, `Args ... args`)
- `void prefetch` (const `int64_t` \*)
- `void getTLBSize` (int &tlb)
- `void queryCacheSizes` (int &l1, int &l2, int &l3)
- `int queryL1CacheSize` ()
- `int queryTopLevelCacheSize` ()

## 17.149 fflas\_pfgemm.inl File Reference

```
#include "fflas-ffpack/paladin/blockcuts.inl"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/paladin/pfgemm_variants.inl"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_pfgemm_INL`
- `#define __FFLASFFPACK_SEQPARTHRESHOLD 220`
- `#define __FFLASFFPACK_DIMKPENALTY 1`

### Functions

- `template<class Field, class ModeTrait, class Strat, class Param >`  
`std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElement<↵`  
`Tag > >::value, typename Field::Element_ptr >::type fgemm` (const `Field` &F, const `FFLAS::FFLAS_TRANSPOSE`  
`ta`, const `FFLAS::FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const  
`typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, `typename`  
`Field::ConstElement_ptr` B, const `size_t` ldb, const `typename Field::Element` beta, `typename Field::Element_ptr`  
`C`, const `size_t` ldc, `MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,`  
`Param > > &H`)

## 17.149.1 Macro Definition Documentation

### 17.149.1.1 \_\_FFLASFFPACK\_fflas\_pfgemm\_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

### 17.149.1.2 \_\_FFLASFFPACK\_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

### 17.149.1.3 \_\_FFLASFFPACK\_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

## 17.150 fflas\_pfttrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_pfttrsm\\_INL](#)
- #define [PTRSM\\_HYBRID\\_THRESHOLD](#) 256

### Functions

- template<class Field , class Cut , class Param >  
[Field::Element\\_ptr](#) ftrsm (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class Field , class Cut , class Param >  
[Field::Element\\_ptr](#) ftrsm (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)

### 17.150.1 Macro Definition Documentation

#### 17.150.1.1 \_\_FFLASFFPACK\_fflas\_pfttrsm\_INL

```
#define __FFLASFFPACK_fflas_pfttrsm_INL
```

#### 17.150.1.2 PTRSM\_HYBRID\_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

## 17.151 fflas\_plevel1.h File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- [FFLAS](#)

## Functions

- `template<class Field >`  
`void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field , class Randlter >`  
`void pfrand (const Field &F, Randlter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field , class Cut , class Param >`  
`Field::Element &fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field , class Cut , class Param >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)`

## 17.152 fflas\_randommatrix.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Namespaces

- **FFPACK**

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- `template<class Field , class Randlter >`  
`Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, Randlter &G)`  
*Random non-zero Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)`  
*Random non-zero Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, Randlter &G)`  
*Random Matrix.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)`  
*Random Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda, Randlter &G)`  
*Random Triangular Matrix.*
- `template<class Field >`  
`Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda)`

*Random Triangular Matrix.*

- `size_t RandInt (size_t a, size_t b)`
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomSymmetricMatrix (const Field &F, size_t n, bool nonsingular, typename`  
`Field::Element_ptr A, size_t lda, RandIter &G)`

*Random Symmetric Matrix.*

- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename`  
`Field::Element_ptr A, size_t lda, RandIter &G)`

*Random Matrix with prescribed rank.*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename`  
`Field::Element_ptr A, size_t lda)`

*Random Matrix with prescribed rank.*

- `size_t * RandomIndexSubset (size_t N, size_t R, size_t *P)`  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- `size_t * RandomPermutation (size_t N, size_t *P)`  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- `void RandomRankProfileMatrix (size_t M, size_t N, size_t R, size_t *rows, size_t *cols)`  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- `void swapval (size_t k, size_t N, size_t *P, size_t val)`
- `void RandomSymmetricRankProfileMatrix (size_t N, size_t R, size_t *rows, size_t *cols)`  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*

- `void RandomLTQSRankProfileMatrix (size_t n, size_t r, size_t t, size_t *rows, size_t *cols)`
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename`  
`Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)`

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename`  
`Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)`

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t N, size_t R, typename`  
`Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)`

*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*

- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R,`  
`typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)`

*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*

- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R,`  
`typename Field::Element_ptr A, size_t lda, RandIter &G)`

*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R,`  
`typename Field::Element_ptr A, size_t lda)`

*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*

- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed det.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed det.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomLTQSMMatrixWithRankandQSorder` (`Field` &F, size\_t n, size\_t r, size\_t t, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)

## 17.153 fflas\_simd.h File Reference

```
#include "fflas-ffpack/utils/fflas_intrinsic.h"
#include <iostream>
#include <type_traits>
#include <limits>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include "givaro/givtypestring.h"
#include <fflas-ffpack/fflas/fflas_simd/simd_modular.inl>
```

### Data Structures

- struct `support_simd< T >`
- struct `is_simd< T >`
- struct `NoSimd< T >`
- struct `SimdChooser< T, bool, bool >`
- struct `SimdChooser< T, false, b >`
- struct `SimdChooser< T, true, false >`
- struct `SimdChooser< T, true, true >`

### Namespaces

- `FFLAS`

### Macros

- `#define SIMD_INT 1`
- `#define INLINE inline`
- `#define CONST`
- `#define PURE`

- `#define` [NORML\\_MOD](#)(C, P, NEGP, MIN, MAX, Q, T)
- `#define` [FLOAT\\_MOD](#)(C, P, INVP, Q)

## Typedefs

- `template<class T >`  
using [Simd](#) = typename [SimdChooser](#)< T >::value

## 17.153.1 Macro Definition Documentation

### 17.153.1.1 SIMD\_INT

```
#define SIMD_INT 1
```

### 17.153.1.2 INLINE

```
#define INLINE inline
```

### 17.153.1.3 CONST

```
#define CONST
```

### 17.153.1.4 PURE

```
#define PURE
```

### 17.153.1.5 NORML\_MOD

```
#define NORML_MOD(  
    C,  
    P,  
    NEGP,  
    MIN,  
    MAX,  
    Q,  
    T )
```

**Value:**

```
{  
    Q = greater(C, MAX);  
    T = lesser(C, MIN);  
    Q = vand(Q, NEGP);  
    T = vand(T, P);  
    Q = vor(Q, T);  
    C = add(C, Q);  
}
```

### 17.153.1.6 FLOAT\_MOD

```
#define FLOAT_MOD(  
    C,
```

```

    P,
    INVP,
    Q )

```

**Value:**

```

{
    Q = mul(C, INVP);
    Q = floor(Q);
    C = fnmadd(C, Q, P);
}

```

## 17.153.2 Typedef Documentation

### 17.153.2.1 Simd

```
using Simd = typename SimdChooser<T>::value
```

## 17.154 fflas\_sparse.C File Reference

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

### 17.154.1 Detailed Description

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

**Author**

Brice Boyer

**See also**

[fflas/fflas\\_sparse.h](#)

## 17.155 fflas\_sparse.h File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/paladin/parallel.h"
#include <recint/recint.h>
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <type_traits>
#include <vector>
#include <iostream>
#include "fflas-ffpack/fflas/fflas_sparse/sparse_matrix_traits.h"
#include "fflas-ffpack/fflas/fflas_sparse/utils.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr.h"
#include "fflas-ffpack/fflas/fflas_sparse/coo.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell.h"
#include "fflas-ffpack/fflas/fflas_sparse/sell.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd.h"

```

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo.h"
#include "fflas-ffpack/fflas/fflas_sparse.inl"
#include "fflas-ffpack/fflas/fflas_sparse/read_sparse.h"
```

## Data Structures

- struct [HelperFlag](#)
- struct [CsrMat< Field >](#)
- struct [CooMat< Field >](#)
- struct [EllMat< Field >](#)
- struct [SpMat< Field, flag >](#)

## Namespaces

- [MKL\\_CONFIG](#)
- [FFLAS](#)
- [FFLAS::sparse\\_details](#)

## Macros

- #define [index\\_t](#) uint32\_t
- #define [ROUND\\_DOWN](#)(x, s) ((x) & ~((s)-1))
- #define [\\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE](#) 64
- #define [assume\\_aligned](#)(pout, pin, v) decltype(pin) pout = pin;
- #define [DENSE\\_THRESHOLD](#) 0.5

## Enumerations

- enum class [SparseMatrix\\_t](#) {  
[CSR](#) , [CSR\\_ZO](#) , [CSC](#) , [CSC\\_ZO](#) ,  
[COO](#) , [COO\\_ZO](#) , [ELL](#) , [ELL\\_ZO](#) ,  
[SELL](#) , [SELL\\_ZO](#) , [ELL\\_simd](#) , [ELL\\_simd\\_ZO](#) ,  
[CSR\\_HYB](#) , [HYB\\_ZO](#) }

## Functions

- template<class Field >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- template<class Field >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- template<class Field , class SM , class FC , class MZO >  
std::enable\_if< !(std::is\_same< typename [ElementTraits](#)< typename [Field::Element](#) >::value, [ElementCategories::MachineFloatTag](#) >::value||std::is\_same< typename [ElementTraits](#)< typename [Field::Element](#) >::value, [ElementCategories::MachineIntTag](#) >::value)>::type [fspmvp\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- template<class Field , class SM , class FC , class MZO >  
std::enable\_if< std::is\_same< typename [ElementTraits](#)< typename [Field::Element](#) >::value, [ElementCategories::MachineFloatTag](#) >::value||std::is\_same< typename [ElementTraits](#)< typename [Field::Element](#) >::value, [ElementCategories::MachineIntTag](#) >::value >::type [fspmvp\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- template<class Field , class SM >  
void [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#), [NotZOSparseMatrix](#))

- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value) >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value) >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, const typename Field::Element &beta, typename Field::Element_ptr y, int Idy)`

## 17.155.1 Macro Definition Documentation

### 17.155.1.1 index\_t

```
#define index_t uint32_t
```

### 17.155.1.2 ROUND\_DOWN

```
#define ROUND_DOWN(  
    x,  
    s ) ((x) & ~((s)-1))
```

### 17.155.1.3 \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

### 17.155.1.4 assume\_aligned

```
#define assume_aligned(  
    pout,
```

```

    pin,
    v ) decltype(pin) pout = pin;

```

### 17.155.1.5 DENSE\_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

## 17.156 fflas\_sparse.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details](#)

### Macros

- `#define \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL`

### Functions

- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if< ! (std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineFloatTag >::value || std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineIntTag >::value) >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if< std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineFloatTag >::value || std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineIntTag >::value >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM >`  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< !isSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typename [Field::Element](#) >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< !isSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typename [Field::Element](#) >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, ZOSparseMatrix)

- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value && support_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< ! (std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value) >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`

- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 17.156.1 Macro Definition Documentation

### 17.156.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

## 17.157 fflas\_transpose.h File Reference

transpose the storage of the matrix (switch between row and col major mode)

```
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
```

## Data Structures

- struct [BlockTransposeSIMD< Field, Simd, >](#)

## Namespaces

- [FFLAS](#)
- [FFLAS::\\_fttranspose\\_impl](#)

## Macros

- `#define FFLAS_TRANSPOSE_BLOCKSIZE 32`
- `#define LD(i) R##i=Simd::loadu(A+lda*i)`
- `#define ST(i) Simd::storeu(B+ldb*i,R##i)`

## Functions

- `template<size_t bs, typename Field , typename BTSimd >`  
`void not_inplace (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`
- `template<size_t bs, typename Field , typename BTSimd >`  
`void square_inplace (const Field &F, const BTSimd &BTS, const size_t m, typename Field::Element_ptr A, const size_t lda)`
- `template<size_t bs, typename Field , typename BTSimd >`  
`void nonsquare_inplace_v1 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)`
- `template<size_t bs, typename Field , typename BTSimd >`  
`void nonsquare_inplace_v2 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)`

### 17.157.1 Detailed Description

transpose the storage of the matrix (switch between row and col major mode)

## 17.157.2 Macro Definition Documentation

### 17.157.2.1 FFLAS\_TRANSPOSE\_BLOCKSIZE

```
#define FFLAS_TRANSPOSE_BLOCKSIZE 32
```

### 17.157.2.2 LD

```
#define LD(  
    i ) R##i=Simd::loadu(A+lda*i)
```

### 17.157.2.3 ST

```
#define ST(  
    i ) Simd::storeu(B+ldb*i,R##i)
```

## 17.158 ffpack-fgesv.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>  
#include <givaro/modular.h>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/fflas_io.h"  
#include <fflas-ffpack/ffpack/ffpack.h>  
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.158.1 Function Documentation

#### 17.158.1.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.159 ffpack-solve.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>  
#include <givaro/modular.h>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/fflas_io.h"  
#include <fflas-ffpack/ffpack/ffpack.h>  
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.159.1 Function Documentation

### 17.159.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

PS: the function Solve will modify the matrix A so here we used a duplicate matrix A2 otherwise  $A \cdot x$  will not be equal to b for the later verification stage

## 17.160 ffpack.C File Reference

C functions calls for [FFPACK](#) in ffpack-c.h.

```
#include "fflas-ffpack/interfaces/libs/ffpack_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const enum [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrsv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)



- [size\\_t pColumnEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [size\\_t pRowEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [size\\_t pReducedColumnEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [size\\_t pReducedRowEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [double \\* Invertin\\_modular\\_double](#) (const double p, const size\_t M, double \*A, const size\_t lda, int \*nullity, bool positive)
- [double \\* Invert\\_modular\\_double](#) (const double p, const size\_t M, const double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- [double \\* Invert2\\_modular\\_double](#) (const double p, const size\_t M, double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- [size\\_t KrylovElim\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const size\_t deg, size\_t \*iterates, size\_t \*inviterates, const size\_t maxit, size\_t virt, bool positive)
- [size\\_t SpecRankProfile\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t deg, size\_t \*rankProfile, bool positive)
- [size\\_t Rank\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- [bool IsSingular\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- [double Det\\_modular\\_double](#) (const double p, const size\_t N, double \*A, const size\_t lda, bool positive)
- [double \\* Solve\\_modular\\_double](#) (const double p, const size\_t M, double \*A, const size\_t lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- [void solveLB\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- [void solveLB2\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- [void RandomNullSpaceVector\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*X, const size\_t incX, bool positive)
- [size\\_t NullSpaceBasis\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*NS, size\_t \*ldn, size\_t \*NSdim, bool positive)
- [size\\_t RowRankProfile\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [size\\_t ColumnRankProfile\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- [void RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag)
- [size\\_t LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)
- [size\\_t RowRankProfileSubmatrixIndices\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- [size\\_t ColRankProfileSubmatrixIndices\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- [size\\_t RowRankProfileSubmatrix\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)
- [size\\_t ColRankProfileSubmatrix\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)
- [void getTriangular\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, bool positive)

- void [getTriangularin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, bool positive)
- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

### 17.160.1 Detailed Description

C functions calls for [FFPACK](#) in `ffpack-c.h`.

Author

Brice Boyer

See also

[ffpack/ffpack.h](#)

### 17.160.2 Function Documentation

#### 17.160.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N )
```

#### 17.160.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N )
```

### 17.160.2.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (
    const double p,
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    bool positive )
```

### 17.160.2.4 PermApplyS\_double()

```
void PermApplyS_double (
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 )
```

### 17.160.2.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (
    const double p,
    double * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    bool positive )
```

### 17.160.2.6 PermApplyT\_double()

```
void PermApplyT_double (
    double * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 )
```

### 17.160.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
    size_t * MathP,
```

```

    const size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N )

```

#### 17.160.2.8 composePermutationsLLL()

```

void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N )

```

#### 17.160.2.9 composePermutationsMLM()

```

void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N )

```

#### 17.160.2.10 cyclic\_shift\_mathPerm()

```

void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s )

```

#### 17.160.2.11 cyclic\_shift\_row\_modular\_double()

```

void cyclic_shift_row_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )

```

#### 17.160.2.12 cyclic\_shift\_col\_modular\_double()

```

void cyclic_shift_col_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )

```

#### 17.160.2.13 applyP\_modular\_double()

```

void applyP_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const enum FFLAS::FFLAS_TRANSPOSE Trans,

```

```
const size_t M,  
const size_t ibeg,  
const size_t iend,  
double * A,  
const size_t lda,  
const size_t * P,  
bool positive )
```

#### 17.160.2.14 fgetrsin\_modular\_double()

```
void fgetrsin_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.160.2.15 fgetrsv\_modular\_double()

```
double* fgetrsv_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.160.2.16 fgesvin\_modular\_double()

```
size_t fgesvin_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,
```

```
int * info,  
bool positive )
```

#### 17.160.2.17 fgesv\_modular\_double()

```
size_t fgesv_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    double * A,  
    const size_t lda,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.160.2.18 ftrtri\_modular\_double()

```
void ftrtri_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_UPLO Uplo,  
    const enum FFLAS::FFLAS_DIAG Diag,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.160.2.19 trinv\_left\_modular\_double()

```
void trinv_left_modular_double (  
    const double p,  
    const size_t N,  
    const double * L,  
    const size_t ldl,  
    double * X,  
    const size_t ldx,  
    bool positive )
```

#### 17.160.2.20 ftrtrm\_modular\_double()

```
void ftrtrm_modular_double (  
    const double p,  
    const FFLAS::FFLAS_SIDE side,  
    const enum FFLAS::FFLAS_DIAG Diag,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

**17.160.2.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    bool positive )
```

**17.160.2.22 LUdivine\_modular\_double()**

```
size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
    const enum FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )
```

**17.160.2.23 ColumnEchelonForm\_modular\_double()**

```
size_t ColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.160.2.24 RowEchelonForm\_modular\_double()**

```
size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.160.2.25 ReducedColumnEchelonForm\_modular\_double()**

```

size_t ReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.160.2.26 ReducedRowEchelonForm\_modular\_double()**

```

size_t ReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.160.2.27 ColumnEchelonForm\_modular\_float()**

```

size_t ColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.160.2.28 RowEchelonForm\_modular\_float()**

```

size_t RowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,

```

```
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

#### 17.160.2.29 ReducedColumnEchelonForm\_modular\_float()

```
size_t ReducedColumnEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.30 ReducedRowEchelonForm\_modular\_float()

```
size_t ReducedRowEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.31 ColumnEchelonForm\_modular\_int32\_t()

```
size_t ColumnEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.32 RowEchelonForm\_modular\_int32\_t()

```
size_t RowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,
```

```

    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.33 ReducedColumnEchelonForm\_modular\_int32\_t()

```

size_t ReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.34 ReducedRowEchelonForm\_modular\_int32\_t()

```

size_t ReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.35 pColumnEchelonForm\_modular\_double()

```

size_t pColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.36 pRowEchelonForm\_modular\_double()

```

size_t pRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,

```

```
size_t * Qt,  
const bool transform,  
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

#### 17.160.2.37 pReducedColumnEchelonForm\_modular\_double()

```
size_t pReducedColumnEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.38 pReducedRowEchelonForm\_modular\_double()

```
size_t pReducedRowEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.39 pColumnEchelonForm\_modular\_float()

```
size_t pColumnEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.160.2.40 pRowEchelonForm\_modular\_float()

```
size_t pRowEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,
```

```

size_t * P,
size_t * Qt,
const bool transform,
const enum FFPACK_C_LU_TAG LuTag,
bool positive )

```

#### 17.160.2.41 pReducedColumnEchelonForm\_modular\_float()

```

size_t pReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.42 pReducedRowEchelonForm\_modular\_float()

```

size_t pReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.43 pColumnEchelonForm\_modular\_int32\_t()

```

size_t pColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.44 pRowEchelonForm\_modular\_int32\_t()

```

size_t pRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,

```

```

    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.45 pReducedColumnEchelonForm\_modular\_int32\_t()

```

size_t pReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.46 pReducedRowEchelonForm\_modular\_int32\_t()

```

size_t pReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.47 Invertin\_modular\_double()

```

double* Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )

```

#### 17.160.2.48 Invert\_modular\_double()

```

double* Invert_modular_double (
    const double p,
    const size_t M,
    const double * A,
    const size_t lda,
    double * X,
    const size_t ldx,

```

```
int * nullity,  
bool positive )
```

#### 17.160.2.49 Invert2\_modular\_double()

```
double* Invert2_modular_double (  
    const double p,  
    const size_t M,  
    double * A,  
    const size_t lda,  
    double * X,  
    const size_t ldx,  
    int * nullity,  
    bool positive )
```

#### 17.160.2.50 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Q,  
    const size_t deg,  
    size_t * iterates,  
    size_t * inviterates,  
    const size_t maxit,  
    size_t virt,  
    bool positive )
```

#### 17.160.2.51 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t deg,  
    size_t * rankProfile,  
    bool positive )
```

#### 17.160.2.52 Rank\_modular\_double()

```
size_t Rank_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

**17.160.2.53 IsSingular\_modular\_double()**

```
bool IsSingular_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.160.2.54 Det\_modular\_double()**

```
double Det_modular_double (
    const double p,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.160.2.55 Solve\_modular\_double()**

```
double* Solve_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * x,
    const int incx,
    const double * b,
    const int incb,
    bool positive )
```

**17.160.2.56 solveLB\_modular\_double()**

```
void solveLB_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.160.2.57 solveLB2\_modular\_double()**

```
void solveLB2_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
```

```
double * L,
const size_t ldl,
const size_t * Q,
double * B,
const size_t ldb,
bool positive )
```

#### 17.160.2.58 RandomNullSpaceVector\_modular\_double()

```
void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )
```

#### 17.160.2.59 NullSpaceBasis\_modular\_double()

```
size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )
```

#### 17.160.2.60 RowRankProfile\_modular\_double()

```
size_t RowRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

#### 17.160.2.61 ColumnRankProfile\_modular\_double()

```
size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
```

```
const enum FFPACK_C_LU_TAG LuTag,
bool positive )
```

### 17.160.2.62 RankProfileFromLU()

```
void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )
```

### 17.160.2.63 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )
```

### 17.160.2.64 RowRankProfileSubmatrixIndices\_modular\_double()

```
size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )
```

### 17.160.2.65 ColRankProfileSubmatrixIndices\_modular\_double()

```
size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )
```

### 17.160.2.66 RowRankProfileSubmatrix\_modular\_double()

```
size_t RowRankProfileSubmatrix_modular_double (
```

```

    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

#### 17.160.2.67 ColRankProfileSubmatrix\_modular\_double()

```

size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

#### 17.160.2.68 getTriangular\_modular\_double()

```

void getTriangular_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )

```

#### 17.160.2.69 getTriangularin\_modular\_double()

```

void getTriangularin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )

```

#### 17.160.2.70 getEchelonForm\_modular\_double()

```

void getEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,

```

```

const enum FFLAS::FFLAS_DIAG Diag,
const size_t M,
const size_t N,
const size_t R,
const size_t * P,
const double * A,
const size_t lda,
double * T,
const size_t ldt,
const bool OnlyNonZeroVectors,
const enum FFPACK_C_LU_TAG LuTag,
bool positive )

```

### 17.160.2.71 getEchelonFormin\_modular\_double()

```

void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.160.2.72 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.160.2.73 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,

```

```

    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.74 getReducedEchelonFormin\_modular\_double()

```

void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.75 getReducedEchelonTransform\_modular\_double()

```

void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.160.2.76 PLUQtoEchelonPermutation()

```

void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )

```

## 17.161 ffpack.doxy File Reference

## 17.162 ffpack.h File Reference

Set of elimination based routines for dense linear algebra.

```

#include "givaro/givpoly1.h"
#include <fflas-ffpack/fflas-ffpack-config.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"

```

```

#include <list>
#include <vector>
#include <iostream>
#include <algorithm>
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "ffpack_fgesv.inl"
#include "ffpack_fgetrs.inl"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
#include "ffpack_pluq.inl"
#include "ffpack_pluq_mp.inl"
#include "ffpack_ppluq.inl"
#include "ffpack_ludivine.inl"
#include "ffpack_ludivine_mp.inl"
#include "ffpack_echelonforms.inl"
#include "ffpack_fsytrf.inl"
#include "ffpack_invert.inl"
#include "ffpack_ftrtr.inl"
#include "ffpack_ftrstr.inl"
#include "ffpack_ftrssyr2k.inl"
#include "ffpack_charpoly_kglu.inl"
#include "ffpack_charpoly_kgfast.inl"
#include "ffpack_charpoly_kgfastgeneralized.inl"
#include "ffpack_charpoly_danilevski.inl"
#include "ffpack_charpoly.inl"
#include "ffpack_frobenius.inl"
#include "ffpack_minpoly.inl"
#include "ffpack_krylovelim.inl"
#include "ffpack_permutation.inl"
#include "ffpack_rankprofiles.inl"
#include "ffpack_det_mp.inl"
#include "ffpack_bruhatgen.inl"
#include "ffpack.inl"

```

## Data Structures

- class [CharpolyFailed](#)

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#) 64
- `#define` [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#) 64

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class Field >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans,

const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P)

*Computes  $P1 \times Diag(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*

- template<class Field >  
void applyP (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Sequential seq)
  - template<class Field , class Cut , class Param >  
void applyP (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
  - template<class Field >  
void MonotonicApplyP (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t R)
- Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class Field >  
void fgetrs (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr B, const size\_t ldb, int \*info)
- Solve the system  $AX = B$  or  $XA = B$ .*
- template<class Field >  
Field::Element\_ptr fgetrs (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr X, const size\_t ldx, typename Field::ConstElement\_ptr B, const size\_t ldb, int \*info)
- Solve the system  $A X = B$  or  $X A = B$ .*
- template<class Field >  
size\_t fgesv (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, int \*info)
- Square system solver.*
- template<class Field >  
size\_t fgesv (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, typename Field::ConstElement\_ptr B, const size\_t ldb, int \*info)
- Rectangular system solver.*
- template<class Field >  
void ftrtri (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG Diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t threshold=\_\_FFLASFFPACK\_FTRTRI\_THRESHOLD)
- Compute the inverse of a triangular matrix.*
- template<class Field >  
void trinv\_left (const Field &F, const size\_t N, typename Field::ConstElement\_ptr L, const size\_t ldl, typename Field::Element\_ptr X, const size\_t ldx)
  - template<class Field >  
void ftrtrm (const Field &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_DIAG diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)
- Compute the product of two triangular matrices of opposite shape.*
- template<class Field >  
void ftrstr (const Field &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diagA, const FFLAS::FFLAS\_DIAG diagB, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSTR\_THRESHOLD)
- Solve a triangular system with a triangular right hand side of the same shape.*

- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field, class Cut, class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field, class Cut, class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PHelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)`  
*Compute the CUP or PLE factorization of the given matrix.*
- `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`

- `template<class Field , class PSHelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`  
`&psH)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Compute the Row Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`  
`&psH)`
- `template<class Field >`  
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Compute the Reduced Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0,`  
`const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`  
`size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Compute the Reduced Row Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const`  
`PSHelper &psH)`
- `template<class Field >`  
`size_t GaussJordan (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const`  
`size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const`  
`FFPACK::FFPACK_LU_TAG LuTag)`  
*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*
- `template<class Field >`  
`Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, int`  
`&nullity)`  
*Invert the given matrix in place or computes its nullity if it is singular.*
- `template<class Field >`  
`Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::ConstElement_ptr A, const size_t`  
`lda, typename Field::Element_ptr X, const size_t idx, int &nullity)`  
*Invert the given matrix or computes its nullity if it is singular.*

- `template<class Field >`  
`Field::Element_ptr Invert2` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` Ida, typename `Field::Element_ptr` X, const `size_t` Idx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly` (const `PolRing` &R, `std::list< typename PolRing::Element > &charp`, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` Ida, typename `PolRing::Domain_t::Randlter` &G, const `FFPACK_CHARPOLY_TAG` CharpTag=`FfpackAuto`, const `size_t` degree=`__FFLASFFPACK_ARITHPROG_THRESHOLD`)  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly` (const `PolRing` &R, typename `PolRing::Element` &charp, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` Ida, typename `PolRing::Domain_t::Randlter` &G, const `FFPACK_CHARPOLY_TAG` CharpTag=`FfpackAuto`, const `size_t` degree=`__FFLASFFPACK_ARITHPROG_THRESHOLD`)  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly` (const `PolRing` &R, typename `PolRing::Element` &charp, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` Ida, const `FFPACK_CHARPOLY_TAG` CharpTag=`FfpackAuto`, const `size_t` degree=`__FFLASFFPACK_ARITHPROG_THRESHOLD`)  
*Compute the characteristic polynomial of the matrix A.*
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KellerGehrig` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` Ida)
- `template<class Field , class Polynomial >`  
`int KGFast` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` Ida, `size_t` \*kg\_mc, `size_t` \*kg\_mb, `size_t` \*kg\_j)
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` Ida)
- `template<class Field >`  
`void fgemv_kgf` (const `Field` &F, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` Ida, typename `Field::ConstElement_ptr` X, const `size_t` incX, typename `Field::Element_ptr` Y, const `size_t` incY, const `size_t` kg\_mc, const `size_t` kg\_mb, const `size_t` kg\_j)
- `template<class Field , class Polynomial , class Randlter >`  
`std::list< Polynomial > & LUKrylov` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` Ida, typename `Field::Element_ptr` U, const `size_t` ldu, `Randlter` &G)
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` Ida)
- `template<class PolRing >`  
`void RandomKrylovPrecond` (const `PolRing` &PR, `std::list< typename PolRing::Element > &completedFactors`, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` Ida, `size_t` &Nb, typename `PolRing::Domain_t::Element_ptr` &B, `size_t` &ldb, typename `PolRing::Domain_t::Randlter` &g, const `size_t` degree=`__FFLASFFPACK_ARITHPROG_THRESHOLD`)
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg` (const `PolRing` &PR, `std::list< typename PolRing::Element > &frobeniusForm`, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` Ida, const `size_t` degree)
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast` (const `Field` &F, `std::list< Polynomial > &charp`, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` Ida, typename `Field::Element_ptr` X, const `size_t` Idx)
- `template<class Field , class Polynomial >`  
`Polynomial & MinPoly` (const `Field` &F, `Polynomial` &minP, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` Ida)  
*Compute the minimal polynomial of the matrix A.*

- template<class Field , class Polynomial , class RandIter >  
Polynomial & **MinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*
- template<class Field , class Polynomial >  
Polynomial & **MatVecMinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class Field , class Polynomial >  
Polynomial & **MatVecMinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr v, const size\_t incv, typename Field::Element\_ptr K, const size\_t ldk, size\_t \*P)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class Field , class Polynomial >  
Polynomial & **Hybrid\_KGF\_LUK\_MinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, size\_t \*P, const FFPACK\_MINPOLY\_TAG MinTag=FFPACK::FfpackDense, const size\_t kg\_mc=0, const size\_t kg\_↔ mb=0, const size\_t kg\_j=0)
- template<class Field >  
size\_t **Rank** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class Field >  
size\_t **pRank** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t numthreads=0)
- template<class Field , class PSHelper >  
size\_t **Rank** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const PSHelper &psH)
- template<class Field >  
bool **IsSingular** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)  
*Returns true if the given matrix is singular.*
- template<class Field >  
Field::Element & **Det** (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- template<class Field >  
Field::Element & **pDet** (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field , class PSHelper >  
Field::Element & **Det** (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field >  
Field::Element\_ptr **Solve** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- template<class Field , class PSHelper >  
Field::Element\_ptr **Solve** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb, PSHelper &psH)
- template<class Field >  
Field::Element\_ptr **pSolve** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb, size\_t numthreads=0)
- template<class Field >  
\*void **RandomNullSpaceVector** (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t incX)

*Solve  $LX = B$  or  $XL = B$  in place.*

- template<class Field >  
size\_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr &NS, size\_t &ldn, size\_t &NSdim)

*Computes a basis of the Left/Right nullspace of the matrix A.*

- template<class Field >  
size\_t RowRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive)

*Computes the row rank profile of A.*

- template<class Field >  
size\_t pRowRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FfpackTileRecursive)

- template<class Field, class PSHelper >  
size\_t RowRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)

- template<class Field >  
size\_t ColumnRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive)

*Computes the column rank profile of A.*

- template<class Field >  
size\_t pColumnRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FfpackTileRecursive)

- template<class Field, class PSHelper >  
size\_t ColumnRankProfile (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)

- void RankProfileFromLU (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag)

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- size\_t LeadingSubmatrixRankProfiles (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template<class Field >  
size\_t RowRankProfileSubmatrixIndices (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*RowRankProfileSubmatrixIndices.*

- template<class Field >  
size\_t ColRankProfileSubmatrixIndices (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*

- template<class Field >  
size\_t RowRankProfileSubmatrix (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr &X, size\_t &R)

*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class Field >  
size\_t ColRankProfileSubmatrix (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr &X, size\_t &R)

*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class Field >  
void getTriangular (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

*Extracts a triangular matrix from a compact storage  $A=L|U$  of rank R.*

- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t Ida)`  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t Ida, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with `transform = true`.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t Ida, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with `transform = true`.*
- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t *P, size_t *Q)`  
*LTBruhatGen Suppose  $A$  is Left Triangular Matrix This procedure computes the Bruhat Representation of  $A$  and return the rank of  $A$ .*
- `template<class Field >`  
`void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)`  
*GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.*

- `template<class Field >`  
`void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt)`  
*GetLTBruhatGen This procedure computes the matrix L or U of the Bruhat Representation Suppose that A is the bruhat representation of a matrix.*
- `size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)`  
*LTQSorder This procedure computes the order of quasiseparability of a matrix.*
- `template<class Field >`  
`size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)`  
*CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.*
- `template<class Field >`  
`void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)`  
*ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.*
- `void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)`  
*Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Representation.*
- `size_t * TInverter (size_t *T, size_t r)`
- `template<class Field >`  
`void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, size_t *MU, size_t *ML)`
- `template<class Field >`  
`void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, const typename Field::Element_ptr Xu, size_t ldu, size_t NbBlocksU, size_t *Ku, size_t *Tu, size_t *MU, const typename Field::Element_ptr XI, size_t ldl, size_t NbBlocksL, size_t *KI, size_t *TI, size_t *ML, typename Field::Element_ptr B, size_t t, size_t ldb, typename Field::Element_ptr C, size_t ldc)`  
*productBruhatxTS Compute the product between the CRE compact representation of a matrix A and B a tall matrix*
- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)`  
*LQUPtoInverseOfFullRankMinor.*

## 17.162.1 Detailed Description

Set of elimination based routines for dense linear algebra.

Matrices are supposed over finite prime field of characteristic less than  $2^{26}$ .

## 17.162.2 Macro Definition Documentation

### 17.162.2.1 \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

### 17.162.2.2 \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

## 17.163 ffpack.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_INL`

### Functions

- `template<class Field >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Computes the rank of the given matrix using a PLUQ factorization.*
- `template<class Field >`  
`size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t numthreads=0)`
- `template<class Field , class PSHelper >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, const PSHelper &psH)`
- `template<class Field >`  
`bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Returns true if the given matrix is singular.*
- `template<class Field >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t *P=NULL, size_t *Q=NULL)`  
*Returns the determinant of the given square matrix.*
- `template<class Field >`  
`Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field , class PSHelper >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)`  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- `template<class Field , class PSHelper >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)`
- `template<class Field >`  
`Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)`
- `template<class Field >`  
`void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr X, const size_t incX)`  
*Solve  $LX = B$  or  $XL = B$  in place.*
- `template<class Field >`  
`size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr &NS, size_t &Idn, size_t &NSdim)`

*Computes a basis of the Left/Right nullspace of the matrix A.*

- template<class Field >  
void [solveLB](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr L, const size\_t ldl, const size\_t \*Q, typename Field::Element\_ptr B, const size\_t ldb)
- template<class Field >  
void [solveLB2](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr L, const size\_t ldl, const size\_t \*Q, typename Field::Element\_ptr B, const size\_t ldb)

## 17.163.1 Macro Definition Documentation

### 17.163.1.1 \_\_FFLASFFPACK\_ffpack\_INL

```
#define __FFLASFFPACK_ffpack_INL
```

## 17.164 ffpack\_bruhatgen.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_bruhatgen\\_inl](#)

### Functions

- template<class Field >  
size\_t [LTBruhatGen](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Q)  
*LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.*
- template<class Field >  
void [getLTBruhatGen](#) (const Field &Fi, const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr R, const size\_t ldr)  
*GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.*
- template<class Field >  
void [getLTBruhatGen](#) (const Field &Fi, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diag, const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr T, const size\_t ldt)  
*GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.*
- size\_t [LTQSorder](#) (const size\_t N, const size\_t r, const size\_t \*P, const size\_t \*Q)  
*LTQSorder This procedure computes the order of quasiseparability of a matrix.*
- template<class Field >  
size\_t [CompressToBlockBiDiagonal](#) (const Field &Fi, const FFLAS::FFLAS\_UPLO Uplo, size\_t N, size\_t s, size\_t r, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr A, size\_t lda, typename Field::Element\_ptr X, size\_t ldx, size\_t \*K, size\_t \*M, size\_t \*T)  
*CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.*

- template<class Field >  
void [ExpandBlockBiDiagonalToBruhat](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_UPLO](#) Uplo, size\_t N, size\_t s, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda, typename [Field::Element\\_ptr](#) X, size\_t ldx, size\_t NbBlocks, size\_t \*K, size\_t \*M, size\_t \*T)  
*ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.*
- void [Bruhat2EchelonPermutation](#) (size\_t N, size\_t R, const size\_t \*P, const size\_t \*Q, size\_t \*M)  
*Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rrepresentation.*
- size\_t \* [TInverter](#) (const size\_t \*T, size\_t r)
- template<class Field >  
void [ComputeRPermutation](#) (const [Field](#) &Fi, size\_t N, size\_t r, const size\_t \*P, const size\_t \*Q, size\_t \*R, const size\_t \*MU, const size\_t \*ML)
- template<class Field >  
[Field::Element\\_ptr](#) [expandLCRE](#) (const [Field](#) &Fi, size\_t N, size\_t s, size\_t r, size\_t \*R, size\_t i, typename [Field::ConstElement\\_ptr](#) Xu, size\_t ldu, size\_t NbBlocksU, const size\_t \*Ku, const size\_t \*Tuinv, typename [Field::ConstElement\\_ptr](#) XI, size\_t ldl, size\_t NbBlocksL, const size\_t \*KI, const size\_t \*Tlinv, typename [Field::Element\\_ptr](#) CRE, size\_t ldcre)  
*Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.*
- template<class Field >  
void [productBruhatxTS](#) (const [Field](#) &Fi, size\_t N, size\_t s, size\_t r, size\_t t, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) Xu, size\_t ldu, size\_t NbBlocksU, const size\_t \*Ku, const size\_t \*Tu, const size\_t \*MU, typename [Field::ConstElement\\_ptr](#) XI, size\_t ldl, size\_t NbBlocksL, const size\_t \*KI, const size\_t \*TI, const size\_t \*ML, typename [Field::Element\\_ptr](#) B, size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) D, size\_t ldd)  
*Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B:  $C \leftarrow A \times B + \text{beta}C$ .*

## 17.164.1 Macro Definition Documentation

### 17.164.1.1 \_\_FFLASFFPACK\_ffpack\_bruhatgen\_inl

```
#define __FFLASFFPACK_ffpack_bruhatgen_inl
```

## 17.165 fpack\_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

### Macros

- #define [FFPACK\\_COMPILED](#)

### Enumerations

- enum [FFLAS\\_C\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 , [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }
- enum [FFLAS\\_C\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 , [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
- enum [FFLAS\\_C\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 , [FflasUpper](#) = 121 , [FflasLower](#) = 122 }
- enum [FFLAS\\_C\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 , [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
- enum [FFLAS\\_C\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 , [FflasLeft](#) = 141 , [FflasRight](#) = 142 }

- enum [FFPACK\\_C\\_LU\\_TAG](#) { [FfpackSlabRecursive](#) = 1 , [FfpackTileRecursive](#) = 2 , [FfpackSingular](#) = 3 }
- enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#) {  
[FfpackLUK](#) =1 , [FfpackKG](#) =2 , [FfpackHybrid](#) =3 , [FfpackKGFast](#) =4 ,  
[FfpackDanilevski](#) =5 , [FfpackArithProg](#) =6 , [FfpackKGFastG](#) =7 }
- enum [FFPACK\\_C\\_MINPOLY\\_TAG](#) { [FfpackDense](#) =1 , [FfpackKGF](#) =2 }

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrs\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void [trinv\\_left\\_modular\\_double](#) (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t ldx, bool positive)
- void [ftrrm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- size\_t [PLUQ\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, bool positive)
- size\_t [LUdivine\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, const size\_t cutoff, bool positive)
- size\_t [LUdivine\\_small\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [LUdivine\\_gauss\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)

- `size_t ColumnEchelonForm_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t RowEchelonForm_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t RowEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t RowEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm2_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, bool positive)
- `size_t REF_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*Qt, size\_t \*P, bool positive)
- `double * Invertin_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const size\_t M, const double \*A, const size\_t lda, double \*X, const size\_t ldx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, double \*X, const size\_t ldx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const size\_t deg, size\_t \*iterates, size\_t \*inviterates, const size\_t maxit, size\_t virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t deg, size\_t \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double Det_modular_double` (const double p, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double * Solve_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*X, const size\_t incX, bool positive)

- `size_t NullSpaceBasis_modular_double` (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*NS, `size_t` \*ldn, `size_t` \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [RankProfileFromLU](#) (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- void [getTriangular\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const `size_t` M, const `size_t` N, const `size_t` R, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, bool positive)
- void [getTriangularin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const `size_t` M, const `size_t` N, const `size_t` R, double \*A, const `size_t` lda, bool positive)
- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, double \*A, const `size_t` lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, double \*A, const `size_t` lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const `size_t` N, const `size_t` R, const `size_t` \*P, `size_t` \*outPerm)

## 17.165.1 Macro Definition Documentation

### 17.165.1.1 FFPACK\_COMPILED

```
#define FFPACK_COMPILED
```

## 17.165.2 Enumeration Type Documentation

### 17.165.2.1 FFLAS\_C\_ORDER

enum [FFLAS\\_C\\_ORDER](#)

#### Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

### 17.165.2.2 FFLAS\_C\_TRANSPOSE

enum [FFLAS\\_C\\_TRANSPOSE](#)

#### Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

### 17.165.2.3 FFLAS\_C\_UPLO

enum [FFLAS\\_C\\_UPLO](#)

#### Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )
FflasUpper	
FflasLower	

### 17.165.2.4 FFLAS\_C\_DIAG

enum [FFLAS\\_C\\_DIAG](#)

#### Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )
FflasNonUnit	
FflasUnit	

### 17.165.2.5 FFLAS\_C\_SIDE

enum [FFLAS\\_C\\_SIDE](#)

## Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.
FflasLeft	
FflasRight	

## 17.165.2.6 FFPACK\_C\_LU\_TAG

enum [FFPACK\\_C\\_LU\\_TAG](#)

## Enumerator

FfpackSlabRecursive	
FfpackTileRecursive	
FfpackSingular	

## 17.165.2.7 FFPACK\_C\_CHARPOLY\_TAG

enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#)

## Enumerator

FfpackLUK	
FfpackKG	
FfpackHybrid	
FfpackKGFast	
FfpackDanilevski	
FfpackArithProg	
FfpackKGFastG	

## 17.165.2.8 FFPACK\_C\_MINPOLY\_TAG

enum [FFPACK\\_C\\_MINPOLY\\_TAG](#)

## Enumerator

FfpackDense	
FfpackKGF	

## 17.165.3 Function Documentation

## 17.165.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
```

```
const size_t * LapackP,  
const size_t N )
```

#### 17.165.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (  
    size_t * LapackP,  
    const size_t * MathP,  
    const size_t N )
```

#### 17.165.3.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (  
    const double p,  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t M2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4,  
    bool positive )
```

#### 17.165.3.4 PermApplyS\_double()

```
void PermApplyS_double (  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t M2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.165.3.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (  
    const double p,  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4,  
    bool positive )
```

#### 17.165.3.6 PermApplyT\_double()

```
void PermApplyT_double (  
    double * A,
```

```
const size_t lda,  
const size_t width,  
const size_t N2,  
const size_t R1,  
const size_t R2,  
const size_t R3,  
const size_t R4 )
```

#### 17.165.3.7 composePermutationsLLM()

```
void composePermutationsLLM (  
    size_t * MathP,  
    const size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.165.3.8 composePermutationsLLL()

```
void composePermutationsLLL (  
    size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.165.3.9 composePermutationsMLM()

```
void composePermutationsMLM (  
    size_t * MathP1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.165.3.10 cyclic\_shift\_mathPerm()

```
void cyclic_shift_mathPerm (  
    size_t * P,  
    const size_t s )
```

#### 17.165.3.11 cyclic\_shift\_row\_modular\_double()

```
void cyclic_shift_row_modular_double (  
    const double p,  
    double * A,  
    size_t m,  
    size_t n,  
    size_t lda,  
    bool positive )
```

#### 17.165.3.12 cyclic\_shift\_col\_modular\_double()

```
void cyclic_shift_col_modular_double (  
    const double p,
```

```
double * A,  
size_t m,  
size_t n,  
size_t lda,  
bool positive )
```

#### 17.165.3.13 applyP\_modular\_double()

```
void applyP_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const enum FFLAS_C_TRANSPOSE Trans,  
    const size_t M,  
    const size_t ibeg,  
    const size_t iend,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    bool positive )
```

#### 17.165.3.14 fgetrsin\_modular\_double()

```
void fgetrsin_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.165.3.15 fgetrs\_modular\_double()

```
double* fgetrs_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

**17.165.3.16 fgesvin\_modular\_double()**

```
size_t fgesvin_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    int * info,
    bool positive )
```

**17.165.3.17 fgesv\_modular\_double()**

```
size_t fgesv_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    const double * B,
    const size_t ldb,
    int * info )
```

**17.165.3.18 ftrtri\_modular\_double()**

```
void ftrtri_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG Diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.165.3.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
    const double p,
    const size_t N,
    const double * L,
    const size_t ldl,
    double * X,
    const size_t ldx,
    bool positive )
```

**17.165.3.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
    const double p,
    const enum FFLAS_C_DIAG diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.165.3.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    bool positive )
```

**17.165.3.22 LUdivine\_modular\_double()**

```
size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )
```

**17.165.3.23 LUdivine\_small\_modular\_double()**

```
size_t LUdivine_small_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.24 LUdivine\_gauss\_modular\_double()**

```
size_t LUdivine_gauss_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.25 ColumnEchelonForm\_modular\_double()**

```
size_t ColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.26 RowEchelonForm\_modular\_double()**

```
size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.27 ColumnEchelonForm\_modular\_float()**

```
size_t ColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.28 RowEchelonForm\_modular\_float()**

```
size_t RowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.29 ColumnEchelonForm\_modular\_int32\_t()**

```
size_t ColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.30 RowEchelonForm\_modular\_int32\_t()**

```
size_t RowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.31 ReducedColumnEchelonForm\_modular\_double()**

```
size_t ReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.32 ReducedRowEchelonForm\_modular\_double()**

```
size_t ReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.33 ReducedColumnEchelonForm\_modular\_float()**

```
size_t ReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.34 ReducedRowEchelonForm\_modular\_float()**

```
size_t ReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.35 ReducedColumnEchelonForm\_modular\_int32\_t()**

```
size_t ReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.36 ReducedRowEchelonForm\_modular\_int32\_t()**

```
size_t ReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.165.3.37 ReducedRowEchelonForm2\_modular\_double()**

```
size_t ReducedRowEchelonForm2_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    bool positive )
```

**17.165.3.38 REF\_modular\_double()**

```
size_t REF_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * Qt,
    size_t * P,
    bool positive )
```

**17.165.3.39 Invertin\_modular\_double()**

```
double* Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )
```

**17.165.3.40 Invert\_modular\_double()**

```
double* Invert_modular_double (
    const double p,
```

```
const size_t M,  
const double * A,  
const size_t lda,  
double * X,  
const size_t ldx,  
int * nullity,  
bool positive )
```

#### 17.165.3.41 Invert2\_modular\_double()

```
double* Invert2_modular_double (  
    const double p,  
    const size_t M,  
    double * A,  
    const size_t lda,  
    double * X,  
    const size_t ldx,  
    int * nullity,  
    bool positive )
```

#### 17.165.3.42 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Q,  
    const size_t deg,  
    size_t * iterates,  
    size_t * inviterates,  
    const size_t maxit,  
    size_t virt,  
    bool positive )
```

#### 17.165.3.43 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t deg,  
    size_t * rankProfile,  
    bool positive )
```

#### 17.165.3.44 Rank\_modular\_double()

```
size_t Rank_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,
```

```
double * A,  
const size_t lda,  
bool positive )
```

#### 17.165.3.45 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (   
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.165.3.46 Det\_modular\_double()

```
double Det_modular_double (   
    const double p,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.165.3.47 Solve\_modular\_double()

```
double* Solve_modular_double (   
    const double p,  
    const size_t M,  
    double * A,  
    const size_t lda,  
    double * x,  
    const int incx,  
    const double * b,  
    const int incb,  
    bool positive )
```

#### 17.165.3.48 solveLB\_modular\_double()

```
void solveLB_modular_double (   
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t R,  
    double * L,  
    const size_t ldl,  
    const size_t * Q,  
    double * B,  
    const size_t ldb )
```

#### 17.165.3.49 solveLB2\_modular\_double()

```
void solveLB2_modular_double (   
    const double p,
```

```

    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )

```

#### 17.165.3.50 RandomNullSpaceVector\_modular\_double()

```

void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )

```

#### 17.165.3.51 NullSpaceBasis\_modular\_double()

```

size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )

```

#### 17.165.3.52 RowRankProfile\_modular\_double()

```

size_t RowRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.165.3.53 ColumnRankProfile\_modular\_double()

```

size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,

```

```

    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.165.3.54 RankProfileFromLU()

```

void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )

```

#### 17.165.3.55 LeadingSubmatrixRankProfiles()

```

size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )

```

#### 17.165.3.56 RowRankProfileSubmatrixIndices\_modular\_double()

```

size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.165.3.57 ColRankProfileSubmatrixIndices\_modular\_double()

```

size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

**17.165.3.58 RowRankProfileSubmatrix\_modular\_double()**

```
size_t RowRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.165.3.59 ColRankProfileSubmatrix\_modular\_double()**

```
size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.165.3.60 getTriangular\_modular\_double()**

```
void getTriangular_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )
```

**17.165.3.61 getTriangularin\_modular\_double()**

```
void getTriangularin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )
```

**17.165.3.62 getEchelonForm\_modular\_double()**

```
void getEchelonForm_modular_double (
```

```

const double p,
const enum FFLAS_C_UPLO Uplo,
const enum FFLAS_C_DIAG diag,
const size_t M,
const size_t N,
const size_t R,
const size_t * P,
const double * A,
const size_t lda,
double * T,
const size_t ldt,
const bool OnlyNonZeroVectors,
const enum FFPACK_C_LU_TAG LuTag,
bool positive )

```

### 17.165.3.63 getEchelonFormin\_modular\_double()

```

void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.165.3.64 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.165.3.65 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,

```

```

    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.165.3.66 getReducedEchelonFormin\_modular\_double()

```

void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.165.3.67 getReducedEchelonTransform\_modular\_double()

```

void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.165.3.68 PLUQtoEchelonPermutation()

```

void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )

```

## 17.166 ffpack\_charpoly.inl File Reference

```

#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "ffpack_charpoly_mp.inl"

```

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_charpoly\\_INL](#)

## Functions

- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain\_t::Element\_ptr A, const size_t Ida, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain\_t::Element\_ptr A, const size_t Ida, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class Field , class Polynomial , class RandIter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t Ida, typename Field::Element\_ptr U, const size_t ldu, RandIter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov\_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t Ida, typename Field::Element\_ptr X, const size_t ldx)`

## 17.166.1 Macro Definition Documentation

### 17.166.1.1 \_\_FFLASFFPACK\_charpoly\_INL

```
#define __FFLASFFPACK_charpoly_INL
```

## 17.167 ffpack\_charpoly\_danilevski.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_danilveski\\_INL](#)

### Functions

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t Ida)`

## 17.167.1 Macro Definition Documentation

## 17.167.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

## 17.168 ffpack\_charpoly\_kgfast.inl File Reference

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfast\\_INL](#)

## Functions

- `template<class Field , class Polynomial >`  
`int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)`
- `template<class Field >`  
`void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement\_ptr A, const size_t lda, type-  
name Field::ConstElement\_ptr X, const size_t incX, typename Field::Element\_ptr Y, const size_t incY, const  
size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

## 17.168.1 Macro Definition Documentation

## 17.168.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

## 17.169 ffpack\_charpoly\_kgfastgeneralized.inl File Reference

```
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfastgeneralized\\_INL](#)

## Functions

- `template<class Field >`  
`Field::Element\_ptr buildMatrix (const Field &F, typename Field::ConstElement\_ptr E, typename Field::ConstElement\_ptr  
C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda,  
const size_t mu)`

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t`  
`N, typename Field::Element_ptr A, const size_t lda)`

## 17.169.1 Macro Definition Documentation

### 17.169.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

## 17.170 fpack\_charpoly\_kglu.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

### Macros

- `#define __FFLASFFPACK_ffpack_charpoly_kglu_INL`

### Functions

- `template<class Field >`  
`size_t updated (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > >`  
`&minpt)`
- `template<class Field >`  
`size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename`  
`Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, type-`  
`name Field::ConstElement_ptr A, const size_t lda)`

## 17.170.1 Macro Definition Documentation

### 17.170.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

## 17.171 fpack\_charpoly\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "givaro/givpoly1.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFPACK_charpoly_mp_INL`

## Functions

- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly` (const `FFPACK::RNSInteger< FFPACK::rns_double > &F`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` charp, const `size_t N`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` A, const `size_t Ida`, `Givaro::ZRing< Givaro::Integer >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, `size_t degree`)
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly` (const `Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R`, `Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp`, const `size_t N`, `Givaro::Integer *A`, const `size_t Ida`, `Givaro::ZRing< Givaro::Integer >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, `size_t degree`)

## 17.171.1 Macro Definition Documentation

### 17.171.1.1 \_\_FFPACK\_charpoly\_mp\_INL

```
#define __FFPACK_charpoly_mp_INL
```

## 17.172 ffpack\_det\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

## Namespaces

- `FFPACK`

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFPACK_det_mp_INL`

## Functions

- `template<class PSHelper > FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det` (const `FFPACK::RNSInteger< FFPACK::rns_double > &F`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` &det, const `size_t N`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` A, const `size_t Ida`, const `PSHelper &psH`)
- `template<class PSHelper > Givaro::Integer & Det` (const `Givaro::ZRing< Givaro::Integer > &F`, `Givaro::Integer &det`, const `size_t N`, `Givaro::Integer *A`, const `size_t Ida`, const `PSHelper &psH`, `size_t *P`, `size_t *Q`)

## 17.172.1 Macro Definition Documentation

### 17.172.1.1 \_\_FFPACK\_det\_mp\_INL

```
#define __FFPACK_det_mp_INL
```

## 17.173 ffpack\_echelonforms.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_echelon_forms_INL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256`

### Functions

- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)`  
*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*
- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)`  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or Column↔EchelonForm.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK↔_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelon↔Form or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG Lu↔Tag=FfpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_↔LU_TAG LuTag=FfpackSlabRecursive)`

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*

- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.*

## 17.173.1 Macro Definition Documentation

### 17.173.1.1 \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

### 17.173.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

## 17.174 ffpack\_fgesv.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fgesv_INL`

### Functions

- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*

## 17.174.1 Macro Definition Documentation

### 17.174.1.1 \_\_FFLASFFPACK\_ffpack\_fgesv\_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

## 17.175 ffpack\_fgetrs.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fgetrs_INL`

### Functions

- `template<class Field >`  
`void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t idx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Solve the system  $A X = B$  or  $X A = B$ .*

### 17.175.1 Macro Definition Documentation

#### 17.175.1.1 \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

## 17.176 ffpack\_frobenius.inl File Reference

```
#include <givaro/givranditer.h>
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFPACK::Protected](#)

### Functions

- `template<class Field >`  
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`

- template<class Field >  
void [DeCompressRowsQK](#) (Field &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t deg, const size\_t nb\_blocs)
- template<class Field >  
void [CompressRowsQA](#) (Field &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class Field >  
void [DeCompressRowsQA](#) (Field &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class PolRing >  
void [RandomKrylovPrecond](#) (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, size\_t &Nb, typename [PolRing::Domain\\_t::Element\\_ptr](#) &B, size\_t &ldb, typename PolRing::Domain\_t::RandIter &g, const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))
- template<class PolRing >  
std::list< typename PolRing::Element > & [ArithProg](#) (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, const size\_t degree)

## 17.177 ffpack\_fsytrf.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_fsytrf\\_INL](#)

### Functions

- template<class Field >  
bool [fsytrf\\_BC\\_Crout](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv)
- template<class Field >  
size\_t [fsytrf\\_BC\\_RL](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv)
- template<class Field >  
size\_t [fsytrf\\_UP\\_RPM\\_BC\\_RL](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class Field >  
size\_t [fsytrf\\_LOW\\_RPM\\_BC\\_Crout](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class Field >  
size\_t [fsytrf\\_UP\\_RPM\\_BC\\_Crout](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class Field >  
size\_t [fsytrf\\_UP\\_RPM](#) (const [Field](#) &Fi, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P, size\_t BCThreshold)
- template<class Field >  
bool [fsytrf\\_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename

```
Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv,
FFLAS::ParSeqHelper::Sequential seq, size_t threshold)
```

- `template<class Field , class Cut , class Param >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv,`  
`FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)`
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)`
- `template<class Field >`  
`void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A,`  
`const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)`

## 17.177.1 Macro Definition Documentation

### 17.177.1.1 \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

## 17.178 ffpack\_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFLASFFPACK_ffpack_ftrssyr2k_INL`

## Functions

- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const`  
`size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t`  
`ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

## 17.178.1 Macro Definition Documentation

## 17.178.1.1 \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

## 17.179 ffpack\_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrstr\\_INL](#)

## Functions

- template<class Field >  
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const [FFLAS::FFLAS\\_DIAG](#) diagB, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#))

*Solve a triangular system with a triangular right hand side of the same shape.*

## 17.179.1 Macro Definition Documentation

## 17.179.1.1 \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

## 17.180 ffpack\_ftrtr.inl File Reference

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1
- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrtr\\_INL](#)

## Functions

- template<class Field >  
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#))  
*Compute the inverse of a triangular matrix.*
- template<class Field >  
void [ftrtrm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)

*Compute the product of two triangular matrices of opposite shape.*

- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)`

## 17.180.1 Macro Definition Documentation

### 17.180.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.180.1.2 \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

## 17.181 ffpack\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

## Macros

- `#define __FFPACK_INST_C`
- `#define FFLAS_COMPILED`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.181.1 Macro Definition Documentation

### 17.181.1.1 \_\_FFPACK\_INST\_C

```
#define __FFPACK_INST_C
```

### 17.181.1.2 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

### 17.181.1.3 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.181.1.4 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.181.1.5 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.181.1.6 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.181.1.7 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.181.1.8 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.181.1.9 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.181.1.10 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.181.1.11 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.182 ffpack\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "ffpack_inst_implem.inl"
```

## Macros

- #define FFLAS\_COMPILED
- #define INST\_OR\_DECL <>
- #define FFLAS\_FIELD Givaro::ModularBalanced
- #define FFLAS\_ELT double
- #define FFLAS\_ELT float

- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.182.1 Macro Definition Documentation

### 17.182.1.1 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

### 17.182.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.182.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.182.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.182.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.182.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.182.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.182.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.182.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.182.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.183 ffpack\_inst\_implem.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<typename Base\_t >  
void [cyclic\\_shift\\_row\\_col](#) (Base\_t \*A, size\_t m, size\_t n, size\_t lda)
- template INST\_OR\_DECL void [cyclic\\_shift\\_row](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template INST\_OR\_DECL void [cyclic\\_shift\\_col](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template INST\_OR\_DECL void [applyP](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P)
- template INST\_OR\_DECL void [fgetrs](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template INST\_OR\_DECL FFLAS\_ELT \* [fgetrs](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*X, const size\_t ldx, const FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template INST\_OR\_DECL size\_t [fgesv](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template INST\_OR\_DECL size\_t [fgesv](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*X, const size\_t ldx, const FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template INST\_OR\_DECL void [ftrtri](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG Diag, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, const size\_t threshold)
- template INST\_OR\_DECL void [trinv\\_left](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, const FFLAS\_ELT \*L, const size\_t ldl, FFLAS\_ELT \*X, const size\_t ldx)
- template INST\_OR\_DECL void [ftrtrm](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_DIAG diag, const size\_t N, FFLAS\_ELT \*A, const size\_t lda)
- template INST\_OR\_DECL size\_t [PLUQ](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template INST\_OR\_DECL size\_t [LUdivine](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const FFLAS::FFLAS\_TRANSPOSE trans, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag, const size\_t cutoff)
- template INST\_OR\_DECL size\_t [LUdivine\\_small](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const FFLAS::FFLAS\_TRANSPOSE trans, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK\_LU\_TAG LuTag)

- template `INST_OR_DECL` `size_t` `LUdivine_gauss` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `RowEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ReducedRowEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ColumnEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ReducedColumnEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` idx, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` idx, int &nullity)
- template `INST_OR_DECL` `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MatVecMinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*V, const `size_t` incv)
- template `INST_OR_DECL` `size_t` `KrylovElim` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt)
- template `INST_OR_DECL` `size_t` `SpecRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile)
- template `INST_OR_DECL` `size_t` `Rank` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` bool `IsSingular` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads >` &parH, `size_t` \*P, `size_t` \*Q)

- template `INST_OR_DECL FFLAS_ELT * Solve` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*x, const `int` incx, const `FFLAS_ELT` \*b, const `int` incb)
- template `INST_OR_DECL void solveLB` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldL, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL void solveLB2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldL, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL void RandomNullSpaceVector` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` incX)
- template `INST_OR_DECL size_t NullSpaceBasis` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&NS, `size_t` &ldn, `size_t` &NSdim)
- template `INST_OR_DECL size_t RowRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL size_t ColumnRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- void `RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- template `INST_OR_DECL size_t RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL size_t ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL size_t RowRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL size_t ColRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors)
- template `INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)

- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- void `PLUQtoEchelonPermutation` (const `size_t` N, const `size_t` R, const `size_t` \*P, `size_t` \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` rank, `FFLAS_ELT` \*A\_factors, const `size_t` lda, const `size_t` \*QtPointer, `FFLAS_ELT` \*X, const `size_t` ldx)

## 17.184 ffpack\_invert.inl File Reference

### Namespaces

- `FFPACK`

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_invert_INL`

### Functions

- template<class Field >  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class Field >  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class Field >  
`Field::Element_ptr Invert2` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*

### 17.184.1 Macro Definition Documentation

#### 17.184.1.1 \_\_FFLASFFPACK\_ffpack\_invert\_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

## 17.185 ffpack\_krylovelim.inl File Reference

### Macros

- `#define __FFLASFFPACK_ffpack_krylovelim_INL`

### 17.185.1 Macro Definition Documentation

## 17.185.1.1 \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

## 17.186 ffpack\_ludivine.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

## Data Structures

- class [callLUdivine\\_small< Element >](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_ludivine\\_INL](#)

## Functions

- [template<class Field >](#)  
[size\\_t LUdivine\\_gauss](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- [template<class Field >](#)  
[size\\_t LUdivine\\_small](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- [template<class Field >](#)  
[size\\_t LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const [size\\_t](#) cutoff)
- [template<class Field >](#)  
[size\\_t LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda, typename [Field::Element\\_ptr](#) X, const [size\\_t](#) idx, typename [Field::Element\\_ptr](#) u, const [size\\_t](#) incu, [size\\_t](#) \*P, bool computeX, const [FFPACK::FFPACK\\_MINPOLY\\_TAG](#) MinTag, const [size\\_t](#) kg\_mc, const [size\\_t](#) kg\_mb, const [size\\_t](#) kg\_j)

## 17.186.1 Macro Definition Documentation

## 17.186.1.1 \_\_FFLASFFPACK\_ffpack\_ludivine\_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

## 17.187 ffpack\_ludivine\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack_ludivine.inl"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFPACK\\_ludivine\\_mp\\_INL](#)

### Functions

- template<> size\_t [LUdivine](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag, const size\_t cutoff)

## 17.187.1 Macro Definition Documentation

### 17.187.1.1 \_\_FFPACK\_ludivine\_mp\_INL

```
#define __FFPACK_ludivine_mp_INL
```

## 17.188 ffpack\_minpoly.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_minpoly\\_INL](#)

### Functions

- template<class Field , class Polynomial >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)  
*Compute the minimal polynomial of the matrix A.*
- template<class Field , class Polynomial , class RandIter >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*

- template<class Field , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class Field , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) v, const size\_t incv, typename [Field::Element\\_ptr](#) K, const size\_t ldk, size\_t \*P)
- template<class Field , class Polynomial >  
Polynomial & [Hybrid\\_KGF\\_LUK\\_MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, size\_t \*P, const FFPACK\_MINPOLY\_TAG MinTag=[FFPACK::FfpackDense](#), const size\_t kg\_mc=0, const size\_t kg\_↵ mb=0, const size\_t kg\_j=0)

## 17.188.1 Macro Definition Documentation

### 17.188.1.1 \_\_FFLASFFPACK\_ffpack\_minpoly\_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

## 17.189 ffpack\_permutation.inl File Reference

```
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

## Namespaces

- [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_permutation\\_INL](#)
- #define [FFLASFFPACK\\_PERM\\_BKSIZE](#) 32

## Functions

- template<class Field >  
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class Field >  
void [MonotonicCompress](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
- template<class Field >  
void [MonotonicCompressMorePivots](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, type-name [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t rowstomove, const size\_t lenP)
- template<class Field >  
void [MonotonicCompressCycles](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t lenP)

- `template<class Field >`  
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)`
- `template<class Field >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)`  
*Conversion of a permutation from LAPACK format to Math format.*
- `void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)`  
*Conversion of a permutation from Maths format to LAPACK format.*
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*

- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes MathP1 x Diag (I\_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.*
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<class Field >  
void [cyclic\\_shift\\_row\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_row](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_row](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_col](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P)  
*Computes P1 x Diag (I\_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.*
- template<class Field, class Cut, class Param >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)

## 17.189.1 Macro Definition Documentation

### 17.189.1.1 \_\_FFLASFFPACK\_ffpack\_permutation\_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

### 17.189.1.2 FFLASFFPACK\_PERM\_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

## 17.190 ffpack\_pluq.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_pluq\\_INL](#)
- #define [CROUT](#)

### Functions

- template<class Field >  
size\_t [PLUQ\\_basecaseV3](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element \*A, const size\_t lda, size\_t \*P, size\_t \*Q)

- `template<class Field >`  
`size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseCrout (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`

*Compute a PLUQ factorization of the given matrix.*

## 17.190.1 Macro Definition Documentation

### 17.190.1.1 \_\_FFLASFFPACK\_ffpack\_pluq\_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

### 17.190.1.2 CROUT

```
#define CROUT
```

## 17.191 ffpack\_pluq\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "givaro/givinteger.h"
#include "givaro/modular-integer.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFPACK_pluq_mp_INL`

## Functions

- `template<class Cut , class Param >`  
`size_t PLUQ (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)`

## 17.191.1 Macro Definition Documentation

### 17.191.1.1 \_\_FFPACK\_pluq\_mp\_INL

```
#define __FFPACK_pluq_mp_INL
```

## 17.192 ffpack\_ppluq.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_ppluq\\_INL](#)
- [#define \\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)
- [#define PBASECASE\\_K 256](#)

### Functions

- [template<class Field >](#)  
void [threads\\_fgemm](#) (const size\_t m, const size\_t n, const size\_t r, int nbthreads, size\_t \*W1, size\_t \*W2, size\_t \*W3, size\_t gamma)
- [template<class Field >](#)  
void [threads\\_ftrsm](#) (const size\_t m, const size\_t n, int nbthreads, size\_t \*t1, size\_t \*t2)
- [template<class Field >](#)  
size\_t [PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads >](#) &PSHelper)
- [template<class Field >](#)  
size\_t [pPLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)

## 17.192.1 Macro Definition Documentation

### 17.192.1.1 \_\_FFLASFFPACK\_ffpack\_ppluq\_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

### 17.192.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

### 17.192.1.3 PBASECASE\_K

```
#define PBASECASE_K 256
```

## 17.193 ffpack\_rankprofiles.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_rank_profiles_INL`

### Functions

- `template<class Field >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Computes the row rank profile of A.*
- `template<class Field >`  
`size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field, class PSHelper >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `template<class Field >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Computes the column rank profile of A.*
- `template<class Field >`  
`size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field, class PSHelper >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_LU_TAG LuTag)`  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- `size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)`  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- `template<class Field >`  
`size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)`  
*RowRankProfileSubmatrixIndices.*
- `template<class Field >`  
`size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)`  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- `template<class Field >`  
`size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- `template<class Field >`  
`size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename`  
`Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X,`  
`const size_t ldx)`  
`LQUPtoInverseOfFullRankMinor.`

## 17.193.1 Macro Definition Documentation

### 17.193.1.1 \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

## 17.194 fgemm\_classical.inl File Reference

```
#include <cmath>
#include "fflas-ffpack/field/field-traits.h"
```

### Macros

- `#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL`

## 17.194.1 Macro Definition Documentation

### 17.194.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL
```

## 17.195 fgemm\_classical\_mp.inl File Reference

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

### Data Structures

- `struct MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >`
- `struct MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`
- `struct MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`

### Namespaces

- `FFLAS`

## Macros

- `#define __FFPACK_fgemm_classical_INL`

## Functions

- `template<typename RNS , typename ParSeqTrait >  
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >  
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`
- `template<typename RNS , typename ParSeqTrait >  
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >  
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >  
Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >  
RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >  
RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`

- `Givaro::Integer * fgemm` (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, const Givaro::Integer \*B, const size\_t ldb, const Givaro::Integer beta, Givaro::Integer \*C, const size\_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper< Algo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- `template<class ParSeq >`  
`Givaro::Integer * fgemm` (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, const Givaro::Integer \*B, const size\_t ldb, const Givaro::Integer beta, Givaro::Integer \*C, const size\_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper< Algo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- `template<size_t K1, size_t K2, class ParSeq >`  
`Reclnt::ruint< K1 > * fgemm` (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > \*A, const size\_t lda, const Reclnt::ruint< K1 > \*B, const size\_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > \*C, const size\_t ldc, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

### 17.195.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

### 17.195.2 Macro Definition Documentation

#### 17.195.2.1 \_\_FFPACK\_fgemm\_classical\_INL

```
#define __FFPACK_fgemm_classical_INL
```

## 17.196 fgemm\_winograd.inl File Reference

```
#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemm_classical.inl"
#include "schedule_winograd.inl"
#include "schedule_winograd_acc.inl"
#include "schedule_winograd_acc_ip.inl"
#include "schedule_winograd_ip.inl"
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_fgemm\\_winograd\\_INL](#)
- `#define` [NEWWINO](#)

## Functions

- `template<class Field >`  
`int WinogradThreshold (const Field &F)`  
*Computes the number of recursive levels to perform.*
- `template<> int WinogradThreshold (const Givaro::Modular< float > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class Field , class FieldMode >`  
`void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class Field , class ModeT >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)`
- `template<class Field , class ModeT , class Cut , class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)`

## 17.196.1 Macro Definition Documentation

### 17.196.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

### 17.196.1.2 NEWWINO

```
#define NEWWINO
```

## 17.197 field-traits.h File Reference

Field Traits.

```
#include <type_traits>
#include "fflas-ffpack/field/rns-double-elt.h"
#include "recint/rmint.h"
#include "givaro/modular-general.h"
#include "givaro/zring.h"
```

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*
- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo< T >](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*
- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: `Givaro::recInt`.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: `GMP`.*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*
- struct [ElementTraits< Element >](#)  
*ElementTraits.*
- struct [ElementTraits< float >](#)
- struct [ElementTraits< double >](#)
- struct [ElementTraits< int8\\_t >](#)
- struct [ElementTraits< int16\\_t >](#)
- struct [ElementTraits< int32\\_t >](#)
- struct [ElementTraits< int64\\_t >](#)
- struct [ElementTraits< uint8\\_t >](#)
- struct [ElementTraits< uint16\\_t >](#)
- struct [ElementTraits< uint32\\_t >](#)
- struct [ElementTraits< uint64\\_t >](#)
- struct [ElementTraits< Givaro::Integer >](#)

- struct [ElementTraits](#)< [RecInt::rint](#)< K > >
- struct [ElementTraits](#)< [RecInt::ruint](#)< K > >
- struct [ElementTraits](#)< [RecInt::rmint](#)< K, MG > >
- struct [ElementTraits](#)< [FFPACK::rns\\_double\\_elt](#) >
- struct [ModeTraits](#)< [Field](#) >

*ModeTraits.*

- struct [ModeTraits](#)< [Givaro::Modular](#)< [Element](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [int64\\_t](#), [uint64\\_t](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [int8\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [int16\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [int32\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [uint8\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [uint16\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [uint32\\_t](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [Givaro::Integer](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [RecInt::ruint](#)< K >, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int8\\_t](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int16\\_t](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int32\\_t](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Givaro::Integer](#) > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< [float](#) > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< [double](#) > >
- struct [ModeTraits](#)< [Givaro::Montgomery](#)< T > >
- struct [FieldTraits](#)< [Field](#) >

*FieldTrait.*

- struct [FieldTraits](#)< [Givaro::ZRing](#)< [RecInt::ruint](#)< K > > >
- struct [FieldTraits](#)< [Givaro::Modular](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [double](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [float](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int16\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint16\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int32\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint32\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int64\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint64\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
- struct [FieldTraits](#)< [FFPACK::RNSInteger](#)< T > >
- struct [FieldTraits](#)< [FFPACK::RNSIntegerMod](#)< T > >
- struct [associatedDelayedField](#)< [Field](#) >
- struct [associatedDelayedField](#)< const [Givaro::Modular](#)< T, X > >
- struct [associatedDelayedField](#)< const [Givaro::ModularBalanced](#)< T > >
- struct [associatedDelayedField](#)< const [Givaro::ZRing](#)< T > >
- struct [associatedDelayedField](#)< const [FFPACK::RNSIntegerMod](#)< RNS > >

## Namespaces

- [Reclnt](#)
- [Givaro](#)
- [FFPACK](#)

*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)
- [FFLAS::FieldCategories](#)

*Traits and categories will need to be placed in a proper file later.*

- [FFLAS::ModeCategories](#)

*Specifies the mode of action for an algorithm w.r.t.*

- [FFLAS::ElementCategories](#)

## Functions

- `template<class Field , class enable = void>  
Field::Residu_t maxCardinality ()`
- `template<> uint64_t maxCardinality< Givaro::Modular< int64_t > > ()`
- `template<> uint32_t maxCardinality< Givaro::Modular< int32_t > > ()`
- `template<class Field >  
Field::Residu_t minCardinality ()`

### 17.197.1 Detailed Description

Field Traits.

## 17.198 field.doxy File Reference

### 17.199 flimits.h File Reference

```
#include <climits>
#include <limits>
#include <type_traits>
#include <givaro/givinteger.h>
```

## Data Structures

- `struct limits< unsigned char >`
- `struct limits< signed char >`
- `struct limits< char >`
- `struct limits< unsigned short int >`
- `struct limits< short int >`
- `struct limits< unsigned int >`
- `struct limits< int >`
- `struct limits< unsigned long >`
- `struct limits< long >`
- `struct limits< unsigned long long >`
- `struct limits< long long >`
- `struct limits< float >`
- `struct limits< double >`
- `struct limits< Givaro::Integer >`
- `struct limits< Reclnt::ruint< K > >`
- `struct limits< Reclnt::rint< K > >`

## Functions

- `template<class T, class E >`  
`std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type` [in\\_range](#) (E e)
- `template<class T, class E >`  
`std::enable_if<(std::is_signed< T >::value) &&! (std::is_signed< E >::value), bool >::type` [in\\_range](#) (E e)
- `template<class T, class E >`  
`std::enable_if<! (std::is_signed< T >::value) &&(std::is_signed< E >::value), bool >::type` [in\\_range](#) (E e)

### 17.199.1 Function Documentation

#### 17.199.1.1 `in_range()` [1/3]

```
std::enable_if<std::is_signed<T>::value == std::is_signed<E>::value, bool>::type in_range (
    E e )
```

#### 17.199.1.2 `in_range()` [2/3]

```
std::enable_if<(std::is_signed<T>::value) && ! (std::is_signed<E>::value), bool>::type in_↵
range (
    E e )
```

#### 17.199.1.3 `in_range()` [3/3]

```
std::enable_if<! (std::is_signed<T>::value) && (std::is_signed<E>::value), bool>::type in_↵
range (
    E e )
```

## 17.200 fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

## Macros

- `#define` [CUBE](#)(x) ((x)\*(x)\*(x))
- `#define` [GFOPS](#)(n, t) ([CUBE](#)(double(n)/1000.0)/(3.0\*t))

## Typedefs

- `typedef` Givaro::Timer [TTimer](#)

## Functions

- `int` [main](#) ()

### 17.200.1 Macro Definition Documentation

### 17.200.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

### 17.200.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.200.2 Typedef Documentation

### 17.200.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.200.3 Function Documentation

### 17.200.3.1 main()

```
int main (  
    void )
```

## 17.201 fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <ctime>
```

### Macros

- #define CUBE(x) ((x)\*(x)\*(x))
- #define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0\*t))

### Typedefs

- typedef Givaro::Timer TTimer

### Functions

- int main ()

## 17.201.1 Macro Definition Documentation

### 17.201.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

### 17.201.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.201.2 Typedef Documentation

### 17.201.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.201.3 Function Documentation

### 17.201.3.1 main()

```
int main (  
    void )
```

## 17.202 ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include <ctime>
```

### Macros

- #define CUBE(x) ((x)\*(x)\*(x))
- #define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0\*t))

### Typedefs

- typedef Givaro::Timer TTimer

### Functions

- int main ()

## 17.202.1 Macro Definition Documentation

### 17.202.1.1 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

### 17.202.1.2 GFOPS

```
#define GFOPS(
    n,
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.202.2 Typedef Documentation

### 17.202.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.202.3 Function Documentation

### 17.202.3.1 main()

```
int main (
    void )
```

## 17.203 hyb\_zo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmmm.inl"
```

## Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::HYB\\_ZO>](#)

## Namespaces

- [FFLAS](#)

## 17.204 hyb\_zo\_pspmm.inl File Reference

## Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_pspmm\\_INL](#)

## Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`

### 17.204.1 Macro Definition Documentation

#### 17.204.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

## 17.205 hyb\_zo\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL`

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)`

### 17.205.1 Macro Definition Documentation

#### 17.205.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

## 17.206 hyb\_zo\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`

### 17.206.1 Macro Definition Documentation

#### 17.206.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 17.207 hyb\_zo\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, uint64_t kmax)`

### 17.207.1 Macro Definition Documentation

#### 17.207.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 17.208 hyb\_zo\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL`

### Functions

- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<typename _Field >`  
`std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)`

### 17.208.1 Macro Definition Documentation

#### 17.208.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

## 17.209 igemm.doxy File Reference

### 17.210 igemm.h File Reference

```
#include "igemm_kernels.h"
#include "igemm_tools.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Enumerations

- `enum number\_kind { zero =0 , one =1 , mone =-1 , other =2 }`

### Functions

- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
`void igemm\_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number\_kind alpha_kind>`  
`void igemm\_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)`

- void [igemm\\_](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*B, const size\_t ldb, const int64\_t beta, int64\_t \*C, const size\_t ldc)

## 17.211 igemm.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_INL](#)

### Functions

- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, int64\_t \*C, size\_t ldc)
- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB, enum number\_kind alpha\_kind>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, int64\_t \*C, size\_t ldc)
- void [igemm](#) (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, const int64\_t beta, int64\_t \*C, size\_t ldc)
- void [igemm\\_](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*B, const size\_t ldb, const int64\_t beta, int64\_t \*C, const size\_t ldc)

### 17.211.1 Macro Definition Documentation

#### 17.211.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

## 17.212 igemm\_kernels.h File Reference

```
#include "igemm_kernels.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Functions

- template<enum number\_kind K>  
void [igebb44](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const int64\_t alpha, const int64\_t \*bIA, const int64\_t \*bIB, int64\_t \*C, size\_t ldc)

- `template<enum number_kind K>`  
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)`

## 17.213 igemm\_kernels.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm_tools.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Macros

- `#define \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL`

### Functions

- `template<enum number_kind K>`  
`void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)`

## 17.213.1 Macro Definition Documentation

### 17.213.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

## 17.214 igemm\_tools.h File Reference

```
#include "igemm_tools.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Functions

- `template<size_t k, bool transpose>`  
void [pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- `template<size_t k, bool transpose>`  
void [pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [gebp](#) (size\_t rows, size\_t cols, size\_t depth, int64\_t \*C, size\_t ldc, const int64\_t \*blockA, size\_t lda, const int64\_t \*BlockB, size\_t ldb, int64\_t \*BlockW)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

## 17.215 igemm\_tools.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL`

### Functions

- `template<size_t k, bool transpose>`  
void [pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- `template<size_t k, bool transpose>`  
void [pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

## 17.215.1 Macro Definition Documentation

### 17.215.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

## 17.216 interfaces.doxy File Reference

## 17.217 kaapi\_routines.inl File Reference

### Macros

- `#define __FFLASFFPACK_KAAPI_ROUTINES_INL`

### 17.217.1 Macro Definition Documentation

#### 17.217.1.1 \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

## 17.218 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`

### Functions

- `int main ()`

### 17.218.1 Macro Definition Documentation

#### 17.218.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 17.218.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.218.2 Function Documentation

#### 17.218.2.1 main()

```
int main (  
    void )
```

## 17.219 mainpage.dox File Reference

### 17.220 Matio.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include "fflas_memory.h"
```

#### Functions

- template<class Field >  
Field::Element\_ptr read\_field (const Field &F, const char \*mat\_file, size\_t \*tni, size\_t \*tnj)
- template<class Field >  
std::ostream & write\_field (const Field &F, std::ostream &c, typename Field::ConstElement\_ptr E, int n, int m, int id, bool mapleFormat=false, bool column\_major=false)

#### 17.220.1 Function Documentation

##### 17.220.1.1 read\_field()

```
Field::Element_ptr read_field (
    const Field & F,
    const char * mat_file,
    size_t * tni,
    size_t * tnj )
```

##### 17.220.1.2 write\_field()

```
std::ostream& write_field (
    const Field & F,
    std::ostream & c,
    typename Field::ConstElement_ptr E,
    int n,
    int m,
    int id,
    bool mapleFormat = false,
    bool column_major = false )
```

### 17.221 matmul.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

#### Functions

- int main (int argc, char \*\*argv)  
*This example computes the matrix multiplication over a defined finite field.*

#### 17.221.1 Function Documentation

### 17.221.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the matrix multiplication over a defined finite field.  
Outputs the product of the matrix given as input.

## 17.222 matmul.doxy File Reference

## 17.223 parallel.h File Reference

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/paladin/blockcuts.inl"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [index\\_t](#) size\_t
- #define [TASK](#)(M, l) {l;}
- #define [WAIT](#)
- #define [CHECK\\_DEPENDENCIES](#)
- #define [BARRIER](#)
- #define [PAR\\_BLOCK](#)
- #define [SYNCH\\_GROUP](#)(Args...) {{Args};}
- #define [THREAD\\_INDEX](#) 0
- #define [NUM\\_THREADS](#) 1
- #define [SET\\_THREADS](#)(num\_threads) {}
- #define [MAX\\_THREADS](#) 1
- #define [READ](#)(Args...)
- #define [WRITE](#)(Args...)
- #define [READWRITE](#)(Args...)
- #define [CONSTREFERENCE](#)(...)
- #define [VALUE](#)(...)
- #define [BEGIN\\_PARALLEL\\_MAIN](#)(Args...) int [main](#)(Args) {
- #define [END\\_PARALLEL\\_MAIN](#)(void) return 0; }
- #define [FORBLOCK1D](#)(iter, m, Helper, Args...)
- #define [FOR1D](#)(i, m, Helper, Args...)
- #define [PARFORBLOCK1D](#)(iter, m, Helper, Args...)
- #define [PARFOR1D](#)(iter, m, Helper, Args...)
- #define [FORBLOCK2D](#)(iter, m, n, Helper, Args...)
- #define [FOR2D](#)(i, j, m, n, Helper, Args...)
- #define [PARFORBLOCK2D](#)(iter, m, n, Helper, Args...) [FORBLOCK2D](#)(iter, m, n, Helper, Args)
- #define [PARFOR2D](#)(i, j, m, n, Helper, Args...) [FOR2D](#)(i, j, m, n, Helper, Args)
- #define [COMMA](#) ,
- #define [MODE](#)(...) \_\_VA\_ARGS\_\_
- #define [RETURNPARAM](#)(f, P1, Args...) P1=f(Args)
- #define [NUMARGS](#)(...) [PP\\_NARG](#)(\_\_VA\_ARGS\_\_,[PP\\_RSEQ\\_N](#)())
- #define [PP\\_NARG](#)(...) [PP\\_ARG\\_N](#)(\_\_VA\_ARGS\_\_)
- #define [PP\\_ARG\\_N](#)( \_1, \_2, \_3, \_4, \_5, \_6, \_7, \_8, \_9, \_10, \_11, \_12, \_13, \_14, \_15, \_16, \_17, \_18, \_19, \_20, \_21, \_22, \_23, \_24, \_25, \_26, \_27, \_28, \_29, \_30, \_31, \_32, \_33, \_34, \_35, \_36, \_37, \_38, \_39, \_40, \_41, \_42, \_43, \_44, \_45, \_46, \_47, \_48, \_49, \_50, \_51, \_52, \_53, \_54, \_55, \_56, \_57, \_58, \_59, \_60, \_61, \_62, \_63, N, ...) N
- #define [PP\\_RSEQ\\_N](#)()

- `#define NOSPLIT() FFLAS::ParSeqHelper::Sequential()`
- `#define splitting_0() FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>()`
- `#define splitting_1(a) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>(a)`
- `#define splitting_2(a, c) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)`
- `#define splitting_3(a, b, c) FFLAS::ParSeqHelper::Parallel<b,c>(a)`
- `#define splitt(_1, _2, _3, NAME, ...) NAME`
- `#define SPLITTER(...) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)`

## 17.223.1 Macro Definition Documentation

### 17.223.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.223.1.2 index\_t

```
#define index_t size_t
```

### 17.223.1.3 TASK

```
#define TASK(
    M,
    I ) {I;}
```

### 17.223.1.4 WAIT

```
#define WAIT
```

### 17.223.1.5 CHECK\_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

### 17.223.1.6 BARRIER

```
#define BARRIER
```

### 17.223.1.7 PAR\_BLOCK

```
#define PAR_BLOCK
```

### 17.223.1.8 SYNCH\_GROUP

```
#define SYNCH_GROUP(
    Args... ) {{Args}};
```

### 17.223.1.9 THREAD\_INDEX

```
#define THREAD_INDEX 0
```

**17.223.1.10 NUM\_THREADS**

```
#define NUM_THREADS 1
```

**17.223.1.11 SET\_THREADS**

```
#define SET_THREADS(  
    num_threads ) {}
```

**17.223.1.12 MAX\_THREADS**

```
#define MAX_THREADS 1
```

**17.223.1.13 READ**

```
#define READ(  
    Args... )
```

**17.223.1.14 WRITE**

```
#define WRITE(  
    Args... )
```

**17.223.1.15 READWRITE**

```
#define READWRITE(  
    Args... )
```

**17.223.1.16 CONSTREFERENCE**

```
#define CONSTREFERENCE(  
    ... )
```

**17.223.1.17 VALUE**

```
#define VALUE(  
    ... )
```

**17.223.1.18 BEGIN\_PARALLEL\_MAIN**

```
#define BEGIN_PARALLEL_MAIN(  
    Args... ) int main(Args) {
```

**17.223.1.19 END\_PARALLEL\_MAIN**

```
#define END_PARALLEL_MAIN(  
    void ) return 0; }
```

**17.223.1.20 FORBLOCK1D**

```
#define FORBLOCK1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy1D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
  decltype(Helper)::Param> iter(m, Helper); \
  for(iter.initialize(); !iter.isTerminated(); ++iter) \
  { Args; } }
```

**17.223.1.21 FOR1D**

```
#define FOR1D(
    i,
    m,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK1D(_internal_iterator, m, Helper, \
  for(auto i=_internal_iterator.begin(); i!=_internal_iterator.end(); ++i) \
  { Args; })
```

**17.223.1.22 PARFORBLOCK1D**

```
#define PARFORBLOCK1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.223.1.23 PARFOR1D**

```
#define PARFOR1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.223.1.24 FORBLOCK2D**

```
#define FORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy2D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
  decltype(Helper)::Param> iter(m,n,Helper); \
  for(iter.initialize(); !iter.isTerminated(); ++iter) \
  { Args; } }
```

**17.223.1.25 FOR2D**

```
#define FOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK2D(_internal_iterator, m, n, Helper,
    for(auto i=_internal_iterator.ibegin(); i!=_internal_iterator.iend(); ++i) \
    for(auto j=_internal_iterator.jbegin(); j!=_internal_iterator.jend(); ++j) \
    { Args; })
```

**17.223.1.26 PARFORBLOCK2D**

```
#define PARFORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... ) FORBLOCK2D(iter, m, n, Helper, Args)
```

**17.223.1.27 PARFOR2D**

```
#define PARFOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... ) FOR2D(i, j, m, n, Helper, Args)
```

**17.223.1.28 COMMA**

```
#define COMMA ,
```

**17.223.1.29 MODE**

```
#define MODE(
    ... ) __VA_ARGS__
```

**17.223.1.30 RETURNPARAM**

```
#define RETURNPARAM(
    f,
    Pl,
    Args... ) Pl=f(Args)
```

**17.223.1.31 NUMARGS**

```
#define NUMARGS(
    ... ) PP_NARG_(__VA_ARGS__, PP_RSEQ_N())
```

### 17.223.1.32 PP\_NARG\_

```
#define PP_NARG_(  
    ... ) PP_ARG_N(__VA_ARGS__)
```

### 17.223.1.33 PP\_ARG\_N

```
#define PP_ARG_N(  
    _1,  
    _2,  
    _3,  
    _4,  
    _5,  
    _6,  
    _7,  
    _8,  
    _9,  
    _10,  
    _11,  
    _12,  
    _13,  
    _14,  
    _15,  
    _16,  
    _17,  
    _18,  
    _19,  
    _20,  
    _21,  
    _22,  
    _23,  
    _24,  
    _25,  
    _26,  
    _27,  
    _28,  
    _29,  
    _30,  
    _31,  
    _32,  
    _33,  
    _34,  
    _35,  
    _36,  
    _37,  
    _38,  
    _39,  
    _40,  
    _41,  
    _42,  
    _43,  
    _44,  
    _45,  
    _46,  
    _47,  
    _48,  
    _49,  
    _50,
```

```

    _51,
    _52,
    _53,
    _54,
    _55,
    _56,
    _57,
    _58,
    _59,
    _60,
    _61,
    _62,
    _63,
    N,
    ... ) N

```

#### 17.223.1.34 PP\_RSEQ\_N

```
#define PP_RSEQ_N( )
```

**Value:**

```

63, 62, 61, 60, \
59, 58, 57, 56, 55, 54, 53, 52, 51, 50, \
49, 48, 47, 46, 45, 44, 43, 42, 41, 40, \
39, 38, 37, 36, 35, 34, 33, 32, 31, 30, \
29, 28, 27, 26, 25, 24, 23, 22, 21, 20, \
19, 18, 17, 16, 15, 14, 13, 12, 11, 10, \
9, 8, 7, 6, 5, 4, 3, 2, 1, 0

```

#### 17.223.1.35 NOSPLIT

```
#define NOSPLIT( ) FFLAS::ParSeqHelper::Sequential()
```

#### 17.223.1.36 splitting\_0

```
#define splitting_0( ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Thread>(a)
```

#### 17.223.1.37 splitting\_1

```
#define splitting_1(
    a ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Thread>(a)
```

#### 17.223.1.38 splitting\_2

```
#define splitting_2(
    a,
    c ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, c>(a)
```

#### 17.223.1.39 splitting\_3

```
#define splitting_3(
    a,
    b,
    c ) FFLAS::ParSeqHelper::Parallel<b, c>(a)
```

## 17.223.1.40 splitt

```
#define splitt(
    _1,
    _2,
    _3,
    NAME,
    ... ) NAME
```

## 17.223.1.41 SPLITTER

```
#define SPLITTER(
    ... ) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0) (↵
__VA_ARGS__)
```

## 17.224 pfgemm\_variants.inl File Reference

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::ConstElement_ptr` B, const `size_t` ldb, const `typename` `Field::Element` beta, `typename` `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, `AlgoT`, `FieldTrait`, `ParSeqHelper::Parallel`< `CuttingStrategy::Block`, `StrategyParameter::Threads` > > &H)
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` AA, const `size_t` lda, const `typename` `Field::ConstElement_ptr` BB, const `size_t` t ldb, const `typename` `Field::Element` beta, `typename` `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, `AlgoT`, `FieldTrait`, `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::ThreeDAdaptive` > > &H)
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` AA, const `size_t` lda, const `typename` `Field::ConstElement_ptr` BB, const `size_t` t ldb, const `typename` `Field::Element` beta, `typename` `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, `AlgoT`, `FieldTrait`, `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::TwoDAdaptive` > > &H)
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` AA, const `size_t` lda, const `typename` `Field::ConstElement_ptr` BB, const `size_t` t ldb, const `typename` `Field::Element` beta, `typename` `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, `AlgoT`, `FieldTrait`, `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::TwoD` > > &H)
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element_ptr pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename` `Field::Element` alpha, const `typename` `Field::ConstElement_ptr` A, const `size_t` lda, const `typename` `Field::ConstElement_ptr` B, const `size_t` ldb, const `typename` `Field::Element` beta, `typename` `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `AlgoT`, `FieldTrait`, `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::ThreeD` > > &H)
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb,

```
const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename
Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb,
const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, Al-
goT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace >
> &H)
```

## 17.225 pfgemv.inl File Reference

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field , class AlgoT , class FieldTrait >`  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, type-  
name [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)
- `template<class Field , class AlgoT , class FieldTrait , class Cut >`  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, type-  
name [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)

## 17.226 pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(m, n, r, t) (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0*double(r)/1000.0*(m+n)/1000)/t`

### Typedefs

- `typedef Givaro::Timer TTimer`

### Functions

- `int main ()`

### 17.226.1 Macro Definition Documentation

### 17.226.1.1 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

### 17.226.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t )  (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0↵
0*double(r)/1000.0*(m+n)/1000)/t
```

## 17.226.2 Typedef Documentation

### 17.226.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.226.3 Function Documentation

### 17.226.3.1 main()

```
int main (
    void )
```

## 17.227 pluq.C File Reference

```
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utis/fflas_io.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.227.1 Function Documentation

### 17.227.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.228 rank.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the rank of a matrix over a defined finite field.*

### 17.228.1 Function Documentation

#### 17.228.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the rank of a matrix over a defined finite field.  
Outputs the rank.

## 17.229 read\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <fstream>
#include <string>
#include <cstdlib>
#include <iterator>
```

### Data Structures

- struct [Coo< Field >](#)
- struct [readMyMachineType< Field, T >](#)
- struct [readMyMachineType< Field, mpz\\_t >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::details\\_spmv](#)

### Macros

- #define [DNS\\_BIN\\_VER](#) 0
- #define [mask\\_t](#) uint64\_t

### Functions

- template<class Field , bool sorted = true, bool read\_integer = false>  
void [readSmsFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, uint64\_t &n nz)
- template<class Field >  
void [readSprFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, uint64\_t &n nz)

- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_floating_point< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()`
- `template<class T >`  
`int getDataType ()`
- `template<class Field >`  
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element\_ptr val, std::ifstream &file, const uint64_t dims, const mask\_t data_type, const mask\_t field_desc)`
- `template<class Field >`  
`void readDnsFormat (const std::string &path, const Field &F, index\_t &rowdim, index\_t &colldim, typename Field::Element\_ptr &val)`
- `template<class Field >`  
`void writeDnsFormat (const std::string &path, const Field &F, const index\_t &rowdim, const index\_t &colldim, typename Field::Element\_ptr A, index\_t ldA)`

## 17.229.1 Macro Definition Documentation

### 17.229.1.1 DNS\_BIN\_VER

```
#define DNS_BIN_VER 0
```

### 17.229.1.2 mask\_t

```
#define mask_t uint64_t
```

## 17.230 regression-check.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
```

## Functions

- bool [check1](#) ()
- bool [check2](#) ()
- bool [check3](#) ()
- bool [check4](#) ()
- bool [checkZeroDimCharpoly](#) ()
- bool [checkZeroDimMinPoly](#) ()
- bool [gf2ModularBalanced](#) ()
- int [main](#) ()

## 17.230.1 Function Documentation

### 17.230.1.1 check1()

```
bool check1 ( )
```

**17.230.1.2 check2()**

```
bool check2 ( )
```

**17.230.1.3 check3()**

```
bool check3 ( )
```

**17.230.1.4 check4()**

```
bool check4 ( )
```

**17.230.1.5 checkZeroDimCharpoly()**

```
bool checkZeroDimCharpoly ( )
```

**17.230.1.6 checkZeroDimMinPoly()**

```
bool checkZeroDimMinPoly ( )
```

**17.230.1.7 gf2ModularBalanced()**

```
bool gf2ModularBalanced ( )
```

**17.230.1.8 main()**

```
int main (
    void )
```

**17.231 rns-double-elt.h File Reference**

rns elt structure with double support

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/cast.h"
```

**Data Structures**

- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)

**Namespaces**

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

**Functions**

- template<> [rns\\_double\\_elt\\_ptr fflas\\_const\\_cast](#) (rns\_double\_elt\_cstptr x)
- template<> [rns\\_double\\_elt\\_cstptr fflas\\_const\\_cast](#) (rns\_double\_elt\_ptr x)

### 17.231.1 Detailed Description

rns elt structure with double support

## 17.232 rns-double-recint.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_double\\_recint\\_INL](#)

### 17.232.1 Macro Definition Documentation

#### 17.232.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

## 17.233 rns-double.h File Reference

rns structure with double support

```
#include <iterator>
#include <vector>
#include <givaro/modular-floating.h>
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include "givaro/modular-extended.h"
#include <recint/ruint.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/field/rns-double-elt.h"
#include "rns-double.inl"
#include "rns-double-recint.inl"
```

### Data Structures

- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter](#)< [RNS](#) >

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)

## Macros

- #define `ROUND_DOWN(x, s) ((x) & ~((s)-1))`

## Functions

- template<> void `fflas_delete` (FFPACK::rns\_double\_elt\_ptr A)
- template<> void `fflas_delete` (FFPACK::rns\_double\_elt\_cstptr A)

### 17.233.1 Detailed Description

rns structure with double support

### 17.233.2 Macro Definition Documentation

#### 17.233.2.1 ROUND\_DOWN

```
#define ROUND_DOWN(
    x,
    s ) ((x) & ~((s)-1))
```

## 17.234 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

## Namespaces

- `FFPACK`

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define `__FFLASFFPACK_field_rns_double_INL`

### 17.234.1 Macro Definition Documentation

#### 17.234.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

## 17.235 rns-integer-mod.h File Reference

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <vector>
#include <cmath>
#include <recint/recint.h>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include <givaro/udl.h>
#include "givaro/modular-extended.h"
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
```

```
#include "fflas-ffpack/fflas/fflas_level1.inl"
#include "fflas-ffpack/fflas/fflas_level2.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/fflas/fflas_fscal_mp.inl"
```

## Data Structures

- class [RNSIntegerMod< RNS >](#)
- class [RNSIntegerMod< RNS >::RandIter](#)

## Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFLAS](#)

## Functions

- [template<> FFPACK::rns\\_double\\_elt\\_ptr fflas\\_new](#) (const [FFPACK::RNSIntegerMod< FFPACK::rns\\_double >](#) &F, const size\_t m, const Alignment align)
- [template<> FFPACK::rns\\_double\\_elt\\_ptr fflas\\_new](#) (const [FFPACK::RNSIntegerMod< FFPACK::rns\\_double >](#) &F, const size\_t m, const size\_t n, const Alignment align)
- [template<typename RNS >](#)  
void [finit\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, size\_t k, const Givaro::Integer \*B, const size\_t ldb, typename [RNS::Element\\_ptr](#) A)
- [template<typename RNS >](#)  
void [finit\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, size\_t k, const Givaro::Integer \*B, const size\_t ldb, typename [RNS::Element\\_ptr](#) A)
- [template<typename RNS >](#)  
void [fconvert\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename [RNS::ConstElement\\_ptr](#) A)
- [template<typename RNS >](#)  
void [fconvert\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename [RNS::ConstElement\\_ptr](#) A)

### 17.235.1 Detailed Description

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.236 rns-integer.h File Reference

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-double.h"
```

## Data Structures

- class [RNSInteger< RNS >](#)
- class [RNSInteger< RNS >::RandIter](#)

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)

## Functions

- `template<> FFPACK::rns\_double\_elt\_ptr fflas\_new (const FFPACK::RNSInteger< FFPACK::rns\_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns\_double\_elt\_ptr fflas\_new (const FFPACK::RNSInteger< FFPACK::rns\_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit\_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert\_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)`

### 17.236.1 Detailed Description

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.237 rns.h File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## 17.238 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_field\_rns\_INL`

### 17.238.1 Macro Definition Documentation

#### 17.238.1.1 [\\_\\_FFLASFFPACK\\_field\\_rns\\_INL](#)

```
#define \_\_FFLASFFPACK\_field\_rns\_INL
```

## 17.239 schedule\_bini.inl File Reference

Bini implementation.

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

## Macros

- `#define __FFLASFFPACK_fgemm_bini_INL`

## Functions

- `template<class Field >`  
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr,`  
`const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A,`  
`const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`  
`typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE`  
`base, const size_t rec_level)`

### 17.239.1 Detailed Description

Bini implementation.

### 17.239.2 Macro Definition Documentation

#### 17.239.2.1 \_\_FFLASFFPACK\_fgemm\_bini\_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

## 17.240 schedule\_winograd.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

## Macros

- `#define __FFLASFFPACK_fgemm_winograd_INL`

## Functions

- `template<class Field , class FieldTrait , class Strat , class Param >`  
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t`  
`mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element`  
`beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Field↵`  
`Trait > &WH)`

### 17.240.1 Macro Definition Documentation

#### 17.240.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

## 17.241 schedule\_winograd\_acc.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_INL`

### Functions

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size_t lda, const typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size_t lda, const typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

### 17.241.1 Macro Definition Documentation

#### 17.241.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

## 17.242 schedule\_winograd\_acc\_ip.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL`

## Functions

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field↔Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelper↔Algo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelper↔Algo::Winograd, FieldTrait > &WH)`

### 17.242.1 Macro Definition Documentation

#### 17.242.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

## 17.243 schedule\_winograd\_ip.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_ip_INL`

## Functions

- `template<class Field , class FieldTrait >`  
`void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelper↔Algo::Winograd, Field↔Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelper↔Algo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename`

[Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelper<  
Algo::Winograd, FieldTrait > &WH)

## 17.243.1 Macro Definition Documentation

### 17.243.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

## 17.244 sell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL\\_ZO>](#)

### Namespaces

- [FFLAS](#)

## 17.245 sell\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_pspmv\\_INL](#)

### Functions

- template<class Field >  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class Field >  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class Field >  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const int64\_t kmax)
- template<class Field >  
void [pfspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class Field >  
void [pfspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class Field >  
void [pfspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)

- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.245.1 Macro Definition Documentation

### 17.245.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

## 17.246 sell\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.246.1 Macro Definition Documentation

### 17.246.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

## 17.247 sell\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo< ValT, IdxT >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::sell\\_details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_utils\\_INL](#)

### Functions

- template<class Field >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL > &A)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A)
- template<class Field >  
void [sparse\\_print](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL > &A)
- template<class Field, class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::SELL > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz, uint64\_t sigma=0)
- template<class Field, class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)

## 17.247.1 Macro Definition Documentation

### 17.247.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

## 17.248 simd.doxy File Reference

## 17.249 simd128.inl File Reference

```
#include "simd128_float.inl"
#include "simd128_double.inl"
```

## Data Structures

- struct [Simd128i\\_base](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_INL](#)

## Typedefs

- template<class T >  
using [Simd128](#) = [Simd128\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

## 17.249.1 Macro Definition Documentation

### 17.249.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

## 17.249.2 Typedef Documentation

### 17.249.2.1 Simd128

```
using Simd128 = Simd128\_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

## 17.250 simd128\_double.inl File Reference

```
#include "givaro/givtypestring.h"  
#include "fflas-ffpack/utils/align-allocator.h"  
#include <vector>  
#include <type_traits>
```

## Data Structures

- struct [Simd128\\_impl](#)< true, false, true, 8 >

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_double\\_INL](#)

## 17.250.1 Macro Definition Documentation

### 17.250.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

## 17.251 simd128\_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd128\\_impl< true, false, true, 4 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_float\\_INL](#)

### 17.251.1 Macro Definition Documentation

#### 17.251.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

## 17.252 simd128\_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd128\\_impl< true, true, true, 2 >](#)
- union [Simd128\\_impl< true, true, true, 2 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 2 >](#)
- union [Simd128\\_impl< true, true, false, 2 >::Converter](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int16\\_INL](#)

### 17.252.1 Macro Definition Documentation

#### 17.252.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

## 17.253 simd128\_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

## Data Structures

- struct [Simd128\\_impl< true, true, true, 4 >](#)
- union [Simd128\\_impl< true, true, true, 4 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 4 >](#)
- union [Simd128\\_impl< true, true, false, 4 >::Converter](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int32\\_INL](#)

### 17.253.1 Macro Definition Documentation

#### 17.253.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

## 17.254 simd128\_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

## Data Structures

- struct [Simd128\\_impl< true, true, true, 8 >](#)
- union [Simd128\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 8 >](#)
- union [Simd128\\_impl< true, true, false, 8 >::Converter](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int64\\_INL](#)
- `#define` [vect\\_t Simd128\\_impl<true,true,true,8>::vect\\_t](#)

### 17.254.1 Macro Definition Documentation

#### 17.254.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

#### 17.254.1.2 vect\_t

```
#define vect_t Simd128\_impl<true,true,true,8>::vect\_t
```

## 17.255 simd256.inl File Reference

```
#include "simd256_float.inl"
#include "simd256_double.inl"
```

## Data Structures

- struct [Simd256fp\\_base](#)
- struct [Simd256i\\_base](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_INL](#)

## Typedefs

- template<class T >  
using [Simd256](#) = [Simd256\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

## 17.255.1 Macro Definition Documentation

### 17.255.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_INL](#)

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

## 17.255.2 Typedef Documentation

### 17.255.2.1 [Simd256](#)

```
using Simd256 = Simd256\_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

## 17.256 [simd256\\_double.inl](#) File Reference

```
#include "givaro/givtypestring.h"  
#include "fflas-ffpack/utils/align-allocator.h"  
#include <vector>  
#include <type_traits>
```

## Data Structures

- struct [Simd256\\_impl](#)< true, false, true, 8 >
- union [Simd256\\_impl](#)< true, false, true, 8 >::Converter

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_double\\_INL](#)

## 17.256.1 Macro Definition Documentation

### 17.256.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_double\\_INL](#)

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

## 17.257 simd256\_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd256\\_impl< true, false, true, 4 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_float\\_INL](#)

### 17.257.1 Macro Definition Documentation

#### 17.257.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

## 17.258 simd256\_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd256\\_impl< true, true, true, 2 >](#)
- union [Simd256\\_impl< true, true, true, 2 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 2 >](#)
- union [Simd256\\_impl< true, true, false, 2 >::Converter](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int16\\_INL](#)

### 17.258.1 Macro Definition Documentation

#### 17.258.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

## 17.259 simd256\_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

## Data Structures

- struct [Simd256\\_impl< true, true, true, 4 >](#)
- union [Simd256\\_impl< true, true, true, 4 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 4 >](#)
- union [Simd256\\_impl< true, true, false, 4 >::Converter](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int32\\_INL](#)

## 17.259.1 Macro Definition Documentation

### 17.259.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

## 17.260 simd256\_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

## Data Structures

- struct [Simd256\\_impl< true, true, true, 8 >](#)
- union [Simd256\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 8 >](#)
- union [Simd256\\_impl< true, true, false, 8 >::Converter](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int64\\_INL](#)
- #define [vect\\_t Simd256\\_impl<true, true, true, 8>::vect\\_t](#)

## 17.260.1 Macro Definition Documentation

### 17.260.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

### 17.260.1.2 vect\_t

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

## 17.261 simd512.inl File Reference

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

### Data Structures

- struct [Simd512i\\_base](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_INL](#)

### Typedefs

- template<class T >  
using [Simd512](#) = [Simd512\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

## 17.261.1 Macro Definition Documentation

### 17.261.1.1 \_\_FFLASFFPACK\_simd512\_INL

```
#define __FFLASFFPACK_simd512_INL
```

## 17.261.2 Typedef Documentation

### 17.261.2.1 Simd512

```
using Simd512 = Simd512\_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

## 17.262 simd512\_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd512\\_impl](#)< true, false, true, 8 >

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_double\\_INL](#)

## 17.262.1 Macro Definition Documentation

**17.262.1.1 \_\_FFLASFFPACK\_simd512\_double\_INL**

```
#define __FFLASFFPACK_simd512_double_INL
```

**17.263 simd512\_float.inl File Reference**

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

**Data Structures**

- struct [Simd512\\_impl](#)< true, false, true, 4 >

**Macros**

- #define [\\_\\_FFLASFFPACK\\_simd512\\_float\\_INL](#)

**17.263.1 Macro Definition Documentation****17.263.1.1 \_\_FFLASFFPACK\_simd512\_float\_INL**

```
#define __FFLASFFPACK_simd512_float_INL
```

**17.264 simd512\_int32.inl File Reference**

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

**Data Structures**

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

**Macros**

- #define [\\_\\_FFLASFFPACK\\_simd512\\_int32\\_INL](#)

**17.264.1 Macro Definition Documentation****17.264.1.1 \_\_FFLASFFPACK\_simd512\_int32\_INL**

```
#define __FFLASFFPACK_simd512_int32_INL
```

## 17.265 simd512\_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

### Data Structures

- struct [Simd512\\_impl< true, true, true, 8 >](#)
- union [Simd512\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd512\\_impl< true, true, false, 8 >](#)
- union [Simd512\\_impl< true, true, false, 8 >::Converter](#)

### Macros

- [#define \\_simd512\\_int64\\_INL](#)
- [#define vect\\_t Simd512\\_impl<true, true, true, 8>::vect\\_t](#)

### 17.265.1 Macro Definition Documentation

#### 17.265.1.1 \_simd512\_int64\_INL

```
#define _simd512_int64_INL
```

#### 17.265.1.2 vect\_t

```
#define vect_t Simd512\_impl<true, true, true, 8>::vect\_t
```

## 17.266 simd\_modular.inl File Reference

### Data Structures

- class [FieldSimd< \\_Field >](#)

## 17.267 solve.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example solve the square system defined by the input over a defined finite field.*

### 17.267.1 Function Documentation

### 17.267.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example solve the quare system defined by the input over a defined finite field.

## 17.268 sparse\_matrix\_traits.h File Reference

```
#include <type_traits>
```

### Data Structures

- struct [isSparseMatrix< Field, M >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_HYB > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::HYB\\_ZO > >](#)
- struct [isZOSparseMatrix< F, M >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO > >](#)
- struct [isSparseMatrixSimdFormat< F, M >](#)
- struct [isSparseMatrixMKLFormat< F, M >](#)
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_plus\\_eq](#)
- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl< C >](#)
- struct [has\\_mul\\_impl< C >](#)
- struct [has\\_mul\\_eq\\_impl< C >](#)
- struct [has\\_plus\\_eq\\_impl< C >](#)
- struct [has\\_minus\\_eq\\_impl< C >](#)
- struct [has\\_minus\\_impl< C >](#)
- struct [has\\_operation< T >](#)

### Namespaces

- [FFLAS](#)

## Typedefs

- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_impl< T > >::type
- template<class T >  
using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_minus\_↵\_impl< T > >::type
- template<class T >  
using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copy\_↵\_assignable< T > >::type
- template<class T >  
using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_↵\_eq\_impl< T > >::type
- template<class T >  
using [has\\_minus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_↵\_minus\_eq\_impl< T > >::type
- template<class T >  
using [has\\_mul](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_impl< T > >::type
- template<class T >  
using [has\\_mul\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_↵\_eq\_impl< T > >::type

## 17.269 test-charpoly-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Macros

- #define [ENABLE\\_CHECKER\\_charpoly](#) 1
- #define [TIME\\_CHECKER\\_CHARPOLY](#) 1

## Functions

- template<class Field , class Polynomial >  
void [printPolynomial](#) (const [Field](#) &F, Polynomial &v)
- int [main](#) (int argc, char \*\*argv)

### 17.269.1 Macro Definition Documentation

### 17.269.1.1 ENABLE\_CHECKER\_charpoly

```
#define ENABLE_CHECKER_charpoly 1
```

### 17.269.1.2 TIME\_CHECKER\_CHARPOLY

```
#define TIME_CHECKER_CHARPOLY 1
```

## 17.269.2 Function Documentation

### 17.269.2.1 printPolynomial()

```
void printPolynomial (
    const Field & F,
    Polynomial & v )
```

### 17.269.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.270 test-charpoly.C File Reference

```
#include <iostream>
#include <iomanip>
#include "givaro/modular.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
#include <chrono>
```

## Functions

- template<class Field , class RandIter >  
bool [launch\\_test](#) (const Field &F, size\_t n, typename Field::Element \*A, size\_t lda, size\_t nbit, RandIter &G, FFPACK::FFPACK\_CHARPOLY\_TAG CT)
- template<class Field >  
bool [run\\_with\\_field](#) (const Givaro::Integer p, uint64\_t bits, size\_t n, std::string file, int variant, size\_t iter, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.270.1 Function Documentation

### 17.270.1.1 launch\_test()

```
bool launch_test (
    const Field & F,
```

```

    size_t n,
    typename Field::Element * A,
    size_t lda,
    size_t nbit,
    RandIter & G,
    FFPACK::FFPACK_CHARPOLY_TAG CT )

```

### 17.270.1.2 run\_with\_field()

```

bool run_with_field (
    const Givaro::Integer p,
    uint64_t bits,
    size_t n,
    std::string file,
    int variant,
    size_t iter,
    uint64_t seed )

```

### 17.270.1.3 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.271 test-compressQ.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <list>
#include <vector>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"

```

## Typedefs

- typedef Givaro::Modular< double > [Field](#)

## Functions

- template<class T >  
std::ostream & [printvect](#) (std::ostream &o, vector< T > &vect)
- int [main](#) (int argc, char \*\*argv)

## 17.271.1 Typedef Documentation

### 17.271.1.1 Field

```

typedef Givaro::Modular<double> Field

```

## 17.271.2 Function Documentation

### 17.271.2.1 printvect()

```
std::ostream& printvect (
    std::ostream & o,
    vector< T > & vect )
```

**Bug** does not belong here

### 17.271.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.272 test-det-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

### Macros

- `#define` [ENABLE\\_CHECKER\\_Det](#) 1
- `#define` [TIME\\_CHECKER\\_Det](#) 1

### Functions

- `int` [main](#) (int argc, char \*\*argv)

## 17.272.1 Macro Definition Documentation

### 17.272.1.1 ENABLE\_CHECKER\_Det

```
#define ENABLE_CHECKER_Det 1
```

### 17.272.1.2 TIME\_CHECKER\_Det

```
#define TIME_CHECKER_Det 1
```

## 17.272.2 Function Documentation

### 17.272.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.273 test-det.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

### Functions

- template<class Field , class RandIter >  
bool [test\\_det](#) (Field &F, size\_t n, int iter, RandIter &G)
- int [main](#) (int argc, char \*\*argv)

### 17.273.1 Function Documentation

#### 17.273.1.1 test\_det()

```
bool test_det (
    Field & F,
    size_t n,
    int iter,
    RandIter & G )
```

**Todo** test with stride

#### 17.273.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.274 test-echelon.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <iomanip>
#include <givaro/modular-balanced.h>
#include <givaro/udl.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include <random>
#include <chrono>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25`
- `#define __FFLASFFPACK_PLUQ_THRESHOLD 25`

## Functions

- `template<class Field , class Randlter >`  
`bool test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_redcoechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`Randlter &G, bool par)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.274.1 Macro Definition Documentation

### 17.274.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.274.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

### 17.274.1.3 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

## 17.274.2 Function Documentation

### 17.274.2.1 test\_colechelon()

```
bool test_colechelon (
    Field & F,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
```

```
FFPACK::FFPACK_LU_TAG LuTag,  
RandIter & G,  
bool par )
```

**Todo** check Ida

#### 17.274.2.2 test\_rowechelon()

```
bool test_rowechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.274.2.3 test\_redcolechelon()

```
bool test_redcolechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.274.2.4 test\_redrowechelon()

```
bool test_redrowechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.274.2.5 run\_with\_field()

```
bool run_with_field (  
    Givaro::Integer q,  
    uint64_t b,
```

```

    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed )

```

### 17.274.2.6 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.275 test-fadd.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"

```

## Functions

- template<class Field >  
bool [test\\_fadd](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_faddin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_fsub](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_fsubin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- int [main](#) (int ac, char \*\*av)

## 17.275.1 Function Documentation

### 17.275.1.1 test\_fadd()

```

bool test_fadd (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.275.1.2 test\_faddin()

```

bool test_faddin (
    const Field & F,
    size_t m,
    size_t k,

```

```

    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.275.1.3 test\_fsub()

```

bool test_fsub (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.275.1.4 test\_fsubin()

```

bool test_fsubin (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.275.1.5 main()

```

int main (
    int ac,
    char ** av )

```

## 17.276 test-fdot.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include <givaro/zring.h>
#include <givaro/modular.h>
#include <random>
#include <chrono>

```

## Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<typename Field >`  
`bool` [check\\_fdot](#) (const [Field](#) &F, size\_t n, typename [Field::ConstElement\\_ptr](#) a, size\_t inca, typename [Field::ConstElement\\_ptr](#) b, size\_t incb)

- `template<class Field >`  
`bool run_with_field` (Givaro::Integer q, size\_t BS, size\_t n, size\_t iters, uint64\_t seed)
- `bool run_with_Integer` (size\_t BS, size\_t n, size\_t iters, uint64\_t seed)
- `int main` (int argc, char \*\*argv)

## 17.276.1 Macro Definition Documentation

### 17.276.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.276.2 Function Documentation

### 17.276.2.1 check\_fdot()

```
bool check_fdot (
    const Field & F,
    size_t n,
    typename Field::ConstElement_ptr a,
    size_t inca,
    typename Field::ConstElement_ptr b,
    size_t incb )
```

### 17.276.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t BS,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.276.2.3 run\_with\_Integer()

```
bool run_with_Integer (
    size_t BS,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.276.2.4 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.277 test-fgemm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<class Field , class RandIter >`  
`bool` [launch\\_MM\\_dispatch](#) (const [Field](#) &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, RandIter &G)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (Givaro::Integer q, uint64\_t b, int m, int n, int k, size\_t iters, uint64\_t seed)
- `int` [main](#) (int argc, char \*\*argv)

## 17.277.1 Macro Definition Documentation

### 17.277.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.277.2 Function Documentation

### 17.277.2.1 launch\_MM\_dispatch()

```
bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    RandIter & G )
```

**Bug** test for ldX equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.277.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int n,
    int k,
    size_t iters,
    uint64_t seed )
```

### 17.277.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.278 test-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
```

### Macros

- #define [ENABLE\\_CHECKER\\_fgemm](#) 1

### Functions

- template<class Field >  
bool [check\\_MM](#) (const [Field](#) &F, const typename [Field::Element\\_ptr](#) Cd, enum [FFLAS\\_TRANSPOSE](#) &ta, enum [FFLAS\\_TRANSPOSE](#) &tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::Element\\_ptr](#) B, size\_t ldb, const typename [Field::Element](#) &beta, const typename [Field::Element\\_ptr](#) C, size\_t ldc)
- template<class Field , class RandIter >  
bool [launch\\_MM](#) (const [Field](#) &F, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t ldc, const size\_t lda, enum [FFLAS\\_TRANSPOSE](#) ta, const size\_t ldb, enum [FFLAS\\_TRANSPOSE](#) tb, size\_t iters, int nbw, bool par, RandIter &G)
- template<class Field , class RandIter >  
bool [launch\\_MM\\_dispatch](#) (const [Field](#) &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, const int nbw, const bool par, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, uint64\_t b, int m, int n, int k, int nbw, size\_t iters, bool par, size\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.278.1 Macro Definition Documentation

### 17.278.1.1 ENABLE\_CHECKER\_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

## 17.278.2 Function Documentation

**17.278.2.1 check\_MM()**

```

bool check_MM (
    const Field & F,
    const typename Field::Element_ptr Cd,
    enum FFLAS_TRANSPOSE & ta,
    enum FFLAS_TRANSPOSE & tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr A,
    size_t lda,
    const typename Field::Element_ptr B,
    size_t ldb,
    const typename Field::Element & beta,
    const typename Field::Element_ptr C,
    size_t ldc )

```

**17.278.2.2 launch\_MM()**

```

bool launch_MM (
    const Field & F,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t ldc,
    const size_t lda,
    enum FFLAS_TRANSPOSE ta,
    const size_t ldb,
    enum FFLAS_TRANSPOSE tb,
    size_t iters,
    int nbw,
    bool par,
    RandIter & G )

```

**17.278.2.3 launch\_MM\_dispatch()**

```

bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const int nbw,
    const bool par,
    RandIter & G )

```

**Bug** test for ldX equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

#### 17.278.2.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int n,
    int k,
    int nbw,
    size_t iters,
    bool par,
    size_t seed )
```

#### 17.278.2.5 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.279 test-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

### Functions

- `template<class Field >`  
`bool check_MV (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, const size_t m, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr X, size_t incX, const typename Field::Element &beta, const typename Field::Element_ptr Y, size_t incY)`
- `template<class Field , class RandIter >`  
`bool launch_MV (const Field &F, const size_t m, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t incX, const size_t incY, size_t iters, bool par, RandIter &G)`
- `template<class Field , class RandIter >`  
`bool launch_MV_dispatch (const Field &F, const int mm, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const bool par, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, int m, int k, size_t iters, bool par, uint64_t seed)`
- `int main (int argc, char **argv)`

#### 17.279.1 Function Documentation

### 17.279.1.1 check\_MV()

```
bool check_MV (
    const Field & F,
    const typename Field::Element_ptr Cd,
    enum FFLAS_TRANSPOSE & ta,
    const size_t m,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr A,
    size_t lda,
    const typename Field::Element_ptr X,
    size_t incX,
    const typename Field::Element & beta,
    const typename Field::Element_ptr Y,
    size_t incY )
```

### 17.279.1.2 launch\_MV()

```
bool launch_MV (
    const Field & F,
    const size_t m,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t lda,
    enum FFLAS_TRANSPOSE ta,
    const size_t incX,
    const size_t incY,
    size_t iters,
    bool par,
    RandIter & G )
```

### 17.279.1.3 launch\_MV\_dispatch()

```
bool launch_MV_dispatch (
    const Field & F,
    const int mm,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const bool par,
    RandIter & G )
```

### 17.279.1.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int k,
    size_t iters,
    bool par,
    uint64_t seed )
```

**17.279.1.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.280 test-fger.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-integral.h>
#include <givaro/modular-balanced.h>
#include <givaro/givintprime.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

**Macros**

- `#define` [TIME](#) 1

**Functions**

- `template<class Field >`  
`bool` [check\\_fger](#) (const [Field](#) &F, const typename [Field::Element\\_ptr](#) Cd, const `size_t` m, const `size_t` n, const  
 typename [Field::Element](#) &alpha, const typename [Field::Element\\_ptr](#) x, const `size_t` incx, const typename  
[Field::Element\\_ptr](#) y, const `size_t` incy, const typename [Field::Element\\_ptr](#) C, const `size_t` ldc)
- `template<class Field , class Randlter >`  
`bool` [launch\\_fger](#) (const [Field](#) &F, const `size_t` m, const `size_t` n, const typename [Field::Element](#) alpha, const  
`size_t` ldc, const `size_t` inca, const `size_t` incb, `size_t` iters, Randlter &G)
- `template<class Field , class Randlter >`  
`bool` [launch\\_fger\\_dispatch](#) (const [Field](#) &F, const `size_t` nn, const typename [Field::Element](#) alpha, const  
`size_t` iters, Randlter &G)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (`int64_t` q, `uint64_t` b, `size_t` n, `size_t` iters, `uint64_t` seed)
- `int` [main](#) (`int` argc, `char **`argv)

**17.280.1 Macro Definition Documentation****17.280.1.1 TIME**

```
#define TIME 1
```

**17.280.2 Function Documentation****17.280.2.1 check\_fger()**

```
bool check_fger (
    const Field & F,
    const typename Field::Element\_ptr Cd,
```

```

    const size_t m,
    const size_t n,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr x,
    const size_t incx,
    const typename Field::Element_ptr y,
    const size_t incy,
    const typename Field::Element_ptr C,
    const size_t ldc )

```

### 17.280.2.2 launch\_fger()

```

bool launch_fger (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const size_t ldc,
    const size_t inca,
    const size_t incb,
    size_t iters,
    RandIter & G )

```

### 17.280.2.3 launch\_fger\_dispatch()

```

bool launch_fger_dispatch (
    const Field & F,
    const size_t nn,
    const typename Field::Element alpha,
    const size_t iters,
    RandIter & G )

```

**Bug** test for incx equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.280.2.4 run\_with\_field()

```

bool run_with_field (
    int64_t q,
    uint64_t b,
    size_t n,
    size_t iters,
    uint64_t seed )

```

### 17.280.2.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.281 test-fgesv.C File Reference

```
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Functions

- template<class Field , class RandIter >  
bool [test\\_square\\_fgesv](#) (Field &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t k, size\_t r, RandIter &G)
- template<class Field , class RandIter >  
bool [test\\_rect\\_fgesv](#) (Field &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t n, size\_t k, size\_t r, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, uint64\_t b, size\_t m, size\_t n, size\_t k, size\_t r, size\_t iters, string fileA, string fileB, uint64\_t &seed)
- int [main](#) (int argc, char \*\*argv)

### 17.281.1 Function Documentation

#### 17.281.1.1 test\_square\_fgesv()

```
bool test_square_fgesv (
    Field & F,
    FFLAS_SIDE side,
    string fileA,
    string fileB,
    size_t m,
    size_t k,
    size_t r,
    RandIter & G )
```

#### 17.281.1.2 test\_rect\_fgesv()

```
bool test_rect_fgesv (
    Field & F,
    FFLAS_SIDE side,
    string fileA,
    string fileB,
    size_t m,
    size_t n,
    size_t k,
    size_t r,
    RandIter & G )
```

#### 17.281.1.3 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
```

```

uint64_t b,
size_t m,
size_t n,
size_t k,
size_t r,
size_t iters,
string fileA,
string fileB,
uint64_t & seed )

```

#### 17.281.1.4 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.282 test-finit.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
#include <random>
#include <chrono>

```

## Functions

- template<class Field >  
bool [test\\_freduce](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t k, size\_t n, size\_t iters, bool timing, uint64\_t seed)
- int [main](#) (int ac, char \*\*av)

### 17.282.1 Function Documentation

#### 17.282.1.1 test\_freduce()

```

bool test_freduce (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.282.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t k,
    size_t n,
    size_t iters,
    bool timing,
    uint64_t seed )
```

### 17.282.1.3 main()

```
int main (
    int ac,
    char ** av )
```

## 17.283 test-fscal.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

## Functions

- template<class Field , class Randlter >  
bool [test\\_fscal](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, Randlter &G)
- template<class Field >  
bool [test\\_fscal](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field , class Randlter >  
bool [test\\_fscaln](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, Randlter &G)
- template<class Field >  
bool [test\\_fscaln](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- int [main](#) (int ac, char \*\*av)

## 17.283.1 Function Documentation

### 17.283.1.1 test\_fscal() [1/2]

```
bool test_fscal (
    const Field & F,
    const typename Field::Element & alpha,
    size_t m,
    size_t k,
    size_t n,
```

```
bool timing,  
RandIter & G )
```

#### 17.283.1.2 test\_fscal() [2/2]

```
bool test_fscal (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

#### 17.283.1.3 test\_fscaln() [1/2]

```
bool test_fscaln (  
    const Field & F,  
    const typename Field::Element & alpha,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    RandIter & G )
```

#### 17.283.1.4 test\_fscaln() [2/2]

```
bool test_fscaln (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

#### 17.283.1.5 main()

```
int main (  
    int ac,  
    char ** av )
```

## 17.284 test-fsyr2k.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iomanip>  
#include <iostream>  
#include <random>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/utils/args-parser.h"  
#include "fflas-ffpack/utils/test-utils.h"  
#include <givaro/modular.h>
```

## Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<typename Field , class RandIter >`  
`bool` [check\\_fsyr2k](#) (const [Field](#) &F, `size_t` n, `size_t` k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (Givaro::Integer q, `size_t` b, `size_t` n, `size_t` k, int a, int c, `size_t` iters, `uint64_t` seed)
- `int` [main](#) (int argc, char \*\*argv)

## 17.284.1 Macro Definition Documentation

### 17.284.1.1 [ENABLE\\_ALL\\_CHECKINGS](#)

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.284.2 Function Documentation

### 17.284.2.1 [check\\_fsyr2k\(\)](#)

```
bool check_fsyr2k (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_TRANSPOSE trans,
    RandIter & Rand )
```

### 17.284.2.2 [run\\_with\\_field\(\)](#)

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    int a,
    int c,
    size_t iters,
    uint64_t seed )
```

### 17.284.2.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.285 test-fsyrr.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename Field , class RandIter >  
bool [check\\_fsyrr](#) (const Field &F, size\_t n, size\_t k, size\_t w, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS::FFLAS\_UPLO uplo, FFLAS::FFLAS\_TRANSPOSE trans, RandIter &Rand)
- template<typename Field , class RandIter >  
bool [check\\_fsyrr\\_diag](#) (const Field &F, size\_t n, size\_t k, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS::FFLAS\_UPLO uplo, FFLAS::FFLAS\_TRANSPOSE trans, RandIter &Rand)
- template<typename Field , class RandIter >  
bool [check\\_fsyrr\\_bkdiag](#) (const Field &F, size\_t n, size\_t k, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS\_UPLO uplo, FFLAS\_TRANSPOSE trans, RandIter &Rand)
- template<class Field , class RandIter >  
bool [check\\_computeS1S2](#) (const Field &F, size\_t N, size\_t K, FFLAS\_TRANSPOSE trans, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t k, size\_t w, int a, int c, size\_t iters, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.285.1 Macro Definition Documentation

### 17.285.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.285.2 Function Documentation

### 17.285.2.1 check\_fsyrr()

```
bool check_fsyrr (
    const Field & F,
    size_t n,
    size_t k,
    size_t w,
    const typename Field::Element & alpha,
```

```

    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )

```

#### 17.285.2.2 check\_fsyrk\_diag()

```

bool check_fsyrk_diag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )

```

#### 17.285.2.3 check\_fsyrk\_bkdiag()

```

bool check_fsyrk_bkdiag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
    RandIter & Rand )

```

#### 17.285.2.4 check\_computeS1S2()

```

bool check_computeS1S2 (
    const Field & F,
    size_t N,
    size_t K,
    FFLAS_TRANSPOSE trans,
    RandIter & G )

```

#### 17.285.2.5 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    size_t w,
    int a,
    int c,
    size_t iters,
    uint64_t seed )

```

#### 17.285.2.6 main()

```

int main (

```

```
int argc,
char ** argv )
```

## 17.286 test-fsytrf.C File Reference

```
#include <iostream>
#include <iterator>
#include <vector>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/utils/test-utils.h"
```

### Functions

- `template<typename T >`  
`std::ostream & operator<< (std::ostream &os, const std::vector< T > &x)`
- `template<class Field , class RandIter >`  
`bool test_RPM_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, size_t r, RandIter &G, size_t threshold)`
- `template<class Field , class RandIter >`  
`bool test_generic_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, RandIter &G, size_t threshold)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t iters, string file, size_t threshold, uint64_t &seed)`
- `int main (int argc, char **argv)`

### 17.286.1 Function Documentation

#### 17.286.1.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & os,
    const std::vector< T > & x )
```

#### 17.286.1.2 test\_RPM\_fsytrf()

```
bool test_RPM_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    size_t r,
    RandIter & G,
    size_t threshold )
```

**17.286.1.3 test\_generic\_fsytrf()**

```
bool test_generic_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    RandIter & G,
    size_t threshold )
```

**17.286.1.4 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t n,
    size_t r,
    size_t iters,
    string file,
    size_t threshold,
    uint64_t & seed )
```

**17.286.1.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.287 test-fftrmm.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

**Macros**

- #define `__FFLASFFPACK_SEQUENTIAL`

**Functions**

- template<typename Field , class RandIter >  
bool `check_fftrmm` (const Field &F, size\_t m, size\_t n, const typename Field::Element &alpha, FFLAS::FFLAS\_SIDE side, FFLAS::FFLAS\_UPLO uplo, FFLAS::FFLAS\_TRANSPOSE trans, FFLAS::FFLAS\_DIAG diag, RandIter &Rand)
- template<class Field >  
bool `run_with_field` (Givaro::Integer q, size\_t b, size\_t m, size\_t n, uint64\_t a, size\_t iters, uint64\_t seed)
- int `main` (int argc, char \*\*argv)

## 17.287.1 Macro Definition Documentation

### 17.287.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

## 17.287.2 Function Documentation

### 17.287.2.1 check\_ffrmv()

```
bool check_ffrmv (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    FFLAS::FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.287.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )
```

### 17.287.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.288 test-ffrmv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <chrono>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<typename Field , class RandIter >`  
`bool check_ftrmv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.288.1 Macro Definition Documentation

### 17.288.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.288.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.288.2 Function Documentation

### 17.288.2.1 check\_ftrmv()

```
bool check_ftrmv (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.288.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.288.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.289 test-ffrsm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.289.1 Macro Definition Documentation

### 17.289.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.289.2 Function Documentation

### 17.289.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.290 test-ffrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<typename Field , class RandIter >`  
`bool check_ftrsm (const Field &F, size_t m, size_t n, const typename Field::Element &alpha,`  
`FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, FFLAS::FFLAS_DIAG`  
`diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t n, uint64_t a, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.290.1 Macro Definition Documentation

### 17.290.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.290.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.290.2 Function Documentation

### 17.290.2.1 check\_ftrsm()

```
bool check_ftrsm (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    FFLAS::FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.290.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )
```

### 17.290.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.291 test-ffrssyr2k.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename Field , class RandIter >  
bool [check\\_ffrssyr2k](#) (const [Field](#) &F, size\_t n, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_DIAG](#) diagA, RandIter &Rand)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.291.1 Macro Definition Documentation

### 17.291.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.291.2 Function Documentation

### 17.291.2.1 check\_ffrssyr2k()

```
bool check_ffrssyr2k (
    const Field & F,
    size_t n,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_DIAG diagA,
    RandIter & Rand )
```

### 17.291.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.291.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.292 test-ftrstr.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

### Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- `template<typename Field , class RandIter >`  
`bool` [check\\_ftrstr](#) (const [Field](#) &F, `size_t` n, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo,  
[FFLAS::FFLAS\\_DIAG](#) diagA, [FFLAS::FFLAS\\_DIAG](#) diagB, RandIter &Rand)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (Givaro::Integer q, `size_t` b, `size_t` n, `size_t` iters, `uint64_t` seed)
- `int` [main](#) (int argc, char \*\*argv)

## 17.292.1 Macro Definition Documentation

### 17.292.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.292.2 Function Documentation

### 17.292.2.1 check\_ftrstr()

```
bool check_ftrstr (
    const Field & F,
    size_t n,
    FFLAS::FFLAS\_SIDE side,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_DIAG diagA,
```

```

    FFLAS::FFLAS_DIAG diagB,
    RandIter & Rand )

```

### 17.292.2.2 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )

```

### 17.292.2.3 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.293 test-ffrsv.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

## Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename Field , class RandIter >  
 bool [check\\_ffrsv](#) (const [Field](#) &F, size\_t n, [FFLAS\\_UPLO](#) uplo, [FFLAS\\_TRANSPOSE](#) trans, [FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class Field >  
 bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.293.1 Macro Definition Documentation

### 17.293.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```

#define __FFLASFFPACK_SEQUENTIAL

```

### 17.293.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.293.2 Function Documentation

### 17.293.2.1 check\_ftrsv()

```
bool check_ftrsv (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.293.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.293.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.294 test-ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Macros

- #define `__FFLASFFPACK_SEQUENTIAL`
- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- template<typename Field , class RandIter >  
bool `check_ftrtri` (const Field &F, size\_t n, FFLAS\_UPLO uplo, FFLAS\_DIAG diag, RandIter &Rand)

- template<class Field >  
bool `run_with_field` (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, uint64\_t seed)
- int `main` (int argc, char \*\*argv)

## 17.294.1 Macro Definition Documentation

### 17.294.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.294.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.294.2 Function Documentation

### 17.294.2.1 check\_ftrtri()

```
bool check_ftrtri (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.294.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.294.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.295 test-interfaces-c.c File Reference

```
#include <fflas-ffpack/interfaces/libs/fflas_c.h>
#include <fflas-ffpack/interfaces/libs/ffpack_c.h>
#include <stdlib.h>
#include <stdio.h>
```

## Functions

- int `main` ()

## 17.295.1 Function Documentation

### 17.295.1.1 main()

```
int main (
    void )
```

## 17.296 test-invert-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

### Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- `int` [main](#) (int argc, char \*\*argv)

## 17.296.1 Macro Definition Documentation

### 17.296.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.296.2 Function Documentation

### 17.296.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.297 test-io.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <random>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Data Structures

- struct [CompactElement< Element >](#)
- struct [CompactElement< double >](#)
- struct [CompactElement< float >](#)
- struct [CompactElement< int64\\_t >](#)
- struct [CompactElement< int32\\_t >](#)
- struct [CompactElement< int16\\_t >](#)

## Functions

- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, uint64\_t b, size\_t m, size\_t n, size\_t iters, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

### 17.297.1 Function Documentation

#### 17.297.1.1 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t iters,
    uint64_t seed )
```

#### 17.297.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.298 test-lu.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

## Macros

- #define [BASECASE\\_K](#) 37
- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [\\_\\_LUDIVINE\\_CUTOFF](#) 1

## Functions

- `template<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans>`  
`bool test_LUdivine (const Field &F, typename Field::ConstElement_ptr A, size_t lda, size_t r, size_t m, size_t n)`  
*Tests the LUdivine routine.*
- `template<class Field, FFLAS_DIAG diag>`  
`bool verifPLUQ (const Field &F, typename Field::ConstElement_ptr A, size_t lda, typename Field::Element_ptr PLUQ, size_t ldpluq, size_t *P, size_t *Q, size_t m, size_t n, size_t R)`  
*Verifies that  $B = PLUQ$  where  $A$  stores  $[L|U]$ .*
- `template<class Field, FFLAS_DIAG diag, class Randlter >`  
`bool test_pluq (const Field &F, typename Field::ConstElement_ptr A, size_t r, size_t m, size_t n, size_t lda, Randlter &G)`  
*Tests the LUdivine routine.*
- `template<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class Randlter >`  
`bool launch_test (const Field &F, size_t r, size_t m, size_t n, Randlter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## Variables

- Givaro::Timer `tperm`
- Givaro::Timer `tgemm`
- Givaro::Timer `tBC`
- Givaro::Timer `ttrsm`
- Givaro::Timer `trest`
- Givaro::Timer `timtot`
- `size_t mvcnt = 0`

## 17.298.1 Macro Definition Documentation

### 17.298.1.1 BASECASE\_K

```
#define BASECASE_K 37
```

### 17.298.1.2 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.298.1.3 \_\_LUDIVINE\_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

## 17.298.2 Function Documentation

**17.298.2.1 test\_LUdivine()**

```
bool test_LUdivine (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    size_t r,
    size_t m,
    size_t n )
```

Tests the LUdivine routine.

**Template Parameters**

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U
<i>Trans</i>	

**Parameters**

<i>F</i>	field
<i>A</i>	Matrix (preallocated)
<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

**Returns**

0 iff correct, 1 otherwise

**17.298.2.2 verifPLUQ()**

```
bool verifPLUQ (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    typename Field::Element_ptr PLUQ,
    size_t ldpluq,
    size_t * P,
    size_t * Q,
    size_t m,
    size_t n,
    size_t R )
```

Verifies that  $B = PLUQ$  where A stores  $[L\backslash U]$ .

**Template Parameters**

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U

**Parameters**

<i>F</i>	field
<i>A</i>	Matrix (preallocated)

**Parameters**

<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

**Returns**

0 iff correct, 1 otherwise

**17.298.2.3 test\_pluq()**

```
bool test_pluq (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t r,
    size_t m,
    size_t n,
    size_t lda,
    RandIter & G )
```

Tests the LUdivine routine.

**Template Parameters**

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U
<i>Trans</i>	

**Parameters**

<i>F</i>	field
<i>A</i>	Matrix (preallocated)
<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

**Returns**

0 iff correct, 1 otherwise

**17.298.2.4 launch\_test()**

```
bool launch_test (
    const Field & F,
    size_t r,
    size_t m,
    size_t n,
    RandIter & G )
```

### 17.298.2.5 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed )
```

### 17.298.2.6 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.298.3 Variable Documentation

### 17.298.3.1 tperm

Givaro::Timer tperm

### 17.298.3.2 tgemm

Givaro::Timer tgemm

### 17.298.3.3 tBC

Givaro::Timer tBC

### 17.298.3.4 ttrsm

Givaro::Timer ttrsm

### 17.298.3.5 trest

Givaro::Timer trest

### 17.298.3.6 timtot

Givaro::Timer timtot

### 17.298.3.7 mvcnt

size\_t mvcnt = 0

## 17.299 test-maxdelayeddim.C File Reference

```
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include <stdlib.h>
#include <stdio.h>
```

### Macros

- #define [MAX\\_WITH\\_SIZE\\_T](#)(x) ( (static\_cast<uint64\_t>(std::numeric\_limits<size\_t>::max()) < x)? std::numeric\_limits<size\_t>::max() : x )

### Functions

- template<class Field >  
bool [test](#) (Givaro::Integer p, size\_t kmax)
- int [main](#) ()

## 17.299.1 Macro Definition Documentation

### 17.299.1.1 MAX\_WITH\_SIZE\_T

```
#define MAX_WITH_SIZE_T(  
    x ) ( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) < x)? std::numeric_limits<size_t>::max() : x )
```

## 17.299.2 Function Documentation

### 17.299.2.1 test()

```
bool test (  
    Givaro::Integer p,  
    size_t kmax )
```

### 17.299.2.2 main()

```
int main (  
    void )
```

## 17.300 test-minpoly.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include <chrono>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
```

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/modular-integer.h>
#include <givaro/givpoly1factor.h>
#include <givaro/givpoly1.h>
```

## Functions

- `template<typename Field , class RandIter >`  
`bool check_minpoly (const Field &F, size_t n, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

### 17.300.1 Function Documentation

#### 17.300.1.1 check\_minpoly()

```
bool check_minpoly (
    const Field & F,
    size_t n,
    RandIter & G )
```

#### 17.300.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

#### 17.300.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.301 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

### 17.302 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

## Functions

- `int main (void)`

## 17.302.1 Function Documentation

### 17.302.1.1 main()

```
int main (
    void )
```

## 17.303 test-nullspace.C File Reference

```
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utis/args-parser.h"
#include "fflas-ffpack/utis/fflas_io.h"
#include "fflas-ffpack/utis/fflas_randommatrix.h"
#include "fflas-ffpack/utis/test-utis.h"
#include "fflas-ffpack/utis/timer.h"
```

## Functions

- template<class Field >  
std::string [checkingMessage](#) (const [Field](#) &F)
- template<class Field >  
[Field::Element\\_ptr](#) [readOrRandomMatrixWithRankAndRandomRPM](#) (const [Field](#) &F, std::string file, size\_t m, size\_t n, size\_t lda, size\_t r, uint64\_t seed)  
*If file is not empty, read it and set m, n, lda and r.*
- template<class Field >  
bool [test\\_nullspace](#) ([Field](#) &F, [FFLAS::FFLAS\\_SIDE](#) side, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, uint64\_t b, size\_t m, size\_t n, size\_t r, size\_t iters, std::string file, uint64\_t &seed)
- int [main](#) (int argc, char \*\*argv)

## 17.303.1 Function Documentation

### 17.303.1.1 checkingMessage()

```
std::string checkingMessage (
    const Field & F )
```

### 17.303.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
Field::Element\_ptr readOrRandomMatrixWithRankAndRandomRPM (
    const Field & F,
    std::string file,
    size_t m,
    size_t n,
    size_t lda,
    size_t r,
    uint64_t seed )
```

If file is not empty, read it and set m, n, lda and r.

Otherwise, generate a random matrix of size m x n with random lda.

### 17.303.1.3 test\_nullspace()

```
bool test_nullspace (
    Field & F,
    FFLAS::FFLAS_SIDE side,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda )
```

### 17.303.1.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    std::string file,
    uint64_t & seed )
```

### 17.303.1.5 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.304 test-permutations.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Functions

- bool [checkMonotonicApplyP](#) (FFLAS\_SIDE Side, FFLAS\_TRANSPOSE trans, size\_t \*P, size\_t N, size\_t R)
- int [main](#) ()

## Variables

- Givaro::Timer [tperm](#)
- Givaro::Timer [tgemm](#)
- Givaro::Timer [tBC](#)
- Givaro::Timer [ttrsm](#)
- Givaro::Timer [trest](#)
- Givaro::Timer [timtot](#)

### 17.304.1 Function Documentation

#### 17.304.1.1 checkMonotonicApplyP()

```
bool checkMonotonicApplyP (
    FFLAS_SIDE Side,
    FFLAS_TRANSPOSE trans,
    size_t * P,
    size_t N,
    size_t R )
```

#### 17.304.1.2 main()

```
int main (
    void )
```

### 17.304.2 Variable Documentation

#### 17.304.2.1 tperm

```
Givaro::Timer tperm
```

#### 17.304.2.2 tgemm

```
Givaro::Timer tgemm
```

#### 17.304.2.3 tBC

```
Givaro::Timer tBC
```

#### 17.304.2.4 ttrsm

```
Givaro::Timer ttrsm
```

#### 17.304.2.5 trest

```
Givaro::Timer trest
```

#### 17.304.2.6 timtot

```
Givaro::Timer timtot
```

### 17.305 test-pluq-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `int` [main](#) (`int argc`, `char **argv`)

### 17.305.1 Macro Definition Documentation

#### 17.305.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.305.2 Function Documentation

#### 17.305.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.306 test-quasisep.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

## Functions

- `template<class Field , FFLAS_DIAG diag, class RandIter >`  
`bool` [test\\_BruhatGenerator](#) (`const` [Field](#) &F, `size_t` n, `size_t` r, `size_t` t, `typename` [Field::ConstElement\\_ptr](#) A, `size_t` lda, `typename` [Field::Element\\_ptr](#) TS, `size_t` l, `RandIter` &G)
- `template<class Field , FFLAS_DIAG diag, class RandIter >`  
`bool` [launch\\_test](#) (`const` [Field](#) &F, `size_t` n, `size_t` r, `size_t` t, `size_t` l, `RandIter` &G)
- `template<class Field , class RandGen >`  
`bool` [testLTQSRPM](#) (`const` [Field](#) &F, `size_t` n, `size_t` r, `size_t` t, `RandGen` &G)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (`Givaro::Integer` q, `uint64_t` b, `size_t` n, `size_t` r, `size_t` t, `size_t` l, `size_t` iters, `uint64_t` seed)
- `int` [main](#) (`int argc`, `char **argv`)

### 17.306.1 Function Documentation

**17.306.1.1 test\_BruhatGenerator()**

```
bool test_BruhatGenerator (
    const Field & F,
    size_t n,
    size_t r,
    size_t t,
    typename Field::ConstElement_ptr A,
    size_t lda,
    typename Field::Element_ptr TS,
    size_t l,
    RandIter & G )
```

**17.306.1.2 launch\_test()**

```
bool launch_test (
    const Field & F,
    size_t n,
    size_t r,
    size_t t,
    size_t l,
    RandIter & G )
```

**17.306.1.3 testLTQSRPM()**

```
bool testLTQSRPM (
    const Field & F,
    size_t n,
    size_t r,
    size_t t,
    RandGen & G )
```

**17.306.1.4 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t n,
    size_t r,
    size_t t,
    size_t l,
    size_t iters,
    uint64_t seed )
```

**17.306.1.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.307 test-rankprofiles.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <chrono>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`

## Functions

- `template<class Field >`  
`bool run_with_field` (Givaro::Integer q, uint64\_t b, size\_t m, size\_t n, size\_t r, size\_t iters, uint64\_t seed, bool par)
- `int main` (int argc, char \*\*argv)

### 17.307.1 Macro Definition Documentation

#### 17.307.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.307.2 Function Documentation

#### 17.307.2.1 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed,
    bool par )
```

#### 17.307.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.308 test-rpm.C File Reference

```
#include <iostream>
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Functions

- bool [checkRPM](#) (size\_t M, size\_t N, size\_t R)
- bool [checkSymmetricRPM](#) (size\_t N, size\_t R)
- int [main](#) (int argc, char \*\*argv)

## 17.308.1 Function Documentation

### 17.308.1.1 [checkRPM\(\)](#)

```
bool checkRPM (
    size_t M,
    size_t N,
    size_t R )
```

### 17.308.1.2 [checkSymmetricRPM\(\)](#)

```
bool checkSymmetricRPM (
    size_t N,
    size_t R )
```

### 17.308.1.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.309 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <array>
#include <vector>
#include <random>
#include <string>
#include <functional>
#include <limits>
#include <type_traits>
#include <algorithm>
```

## Data Structures

- struct [ALL](#)< true, v... >
- struct [ALL](#)< false, v... >
- struct [ALL](#)<>

- struct [count\\_nonconst\\_lvalue\\_reference](#)< T, O... >
- struct [count\\_nonconst\\_lvalue\\_reference](#)< T &, O... >
- struct [count\\_nonconst\\_lvalue\\_reference](#)< const T &, O... >
- struct [count\\_nonconst\\_lvalue\\_reference](#)<>
- struct [is\\_all\\_same](#)< T, Args... >
- struct [is\\_all\\_same](#)<>
- struct [width](#)< T >
- struct [width](#)< float >
- struct [width](#)< double >
- class [TestOneMethod](#)< Simd >
- struct [ScalFunctionsBase](#)< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >
- class [ScalFunctionsBase](#)< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >::FloatingPointTestD
- struct [ScalFunctionsBase](#)< Element, typename enable\_if< is\_integral< Element >::value >::type >
- struct [ScalFunctions](#)< Element >

## Macros

- [#define \\_TEST\\_ONE](#)(K, f1, f2, r, n)
- [#define TEST\\_ONE\\_OP](#)(f)
- [#define TEST\\_ONE\\_OP\\_WZ](#)(f)
- [#define TEST\\_IMPL](#)(SIZE, Elt)

## Functions

- [template](#)<typename Element >  
[enable\\_if](#)< is\_integral< Element >::value, bool >::type [check\\_eq](#) (Element x, Element y)
- [template](#)<typename Element >  
[enable\\_if](#)< is\_floating\_point< Element >::value, bool >::type [check\\_eq](#) (Element x, Element y)
- [template](#)<typename Element >  
[bool cmp](#) (vector< Element > out\_scal, vector< Element > out\_simd)
- [template](#)<typename Ret , typename T >  
[Ret eval\\_func\\_on\\_array](#) (function< Ret()> f, array< T, 0 > &arr)
- [template](#)<typename T , typename... TArgs>  
[void eval\\_func\\_on\\_array](#) (function< void(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- [template](#)<typename Ret , typename T , typename... TArgs>  
[Ret eval\\_func\\_on\\_array](#) (function< Ret(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- [template](#)<typename E >  
[std::ostream & operator<<](#) (std::ostream &o, const vector< E > &V)
- [template](#)<typename Simd , typename Element >  
[enable\\_if](#)< is\_floating\_point< Element >::value, bool >::type [test\\_impl\\_base](#) ()
- [template](#)<typename Simd , typename Element >  
[enable\\_if](#)< is\_integral< Element >::value, bool >::type [test\\_impl\\_base](#) ()
- [template](#)<typename Simd , typename Element >  
[bool test\\_impl](#) ()
- [int main](#) (int argc, char \*argv[])

### 17.309.1 Macro Definition Documentation

**17.309.1.1 \_TEST\_ONE**

```
#define _TEST_ONE(
    K,
    f1,
    f2,
    r,
    n )
```

**Value:**

```
do { \
    K T(f1, f2, r, n); \
    bool b = T.writeResultLine(); \
    if (b == false) \
        T.writeDebugData(); \
    btest &= b; \
} while (0)
```

**17.309.1.2 TEST\_ONE\_OP**

```
#define TEST_ONE_OP(
    f )
```

**Value:**

```
_TEST_ONE(TestOneMethod<Simd>, \
function<decltype(Simd::f)>(Simd::f), \
function<decltype(Scal::f)>(Scal::f), \
function<decltype(Scal::genInputs)>(Scal::genInputs), #f)
```

**17.309.1.3 TEST\_ONE\_OP\_WZ**

```
#define TEST_ONE_OP_WZ(
    f )
```

**Value:**

```
_TEST_ONE(TestOneMethod<Simd>, \
function<decltype(Simd::f)>(Simd::f), \
function<decltype(Scal::f)>(Scal::f), \
function<decltype(Scal::genInputsWithZero)>(Scal::genInputsWithZero), \
#f " test with zero")
```

**17.309.1.4 TEST\_IMPL**

```
#define TEST_IMPL(
    SIZE,
    Elt )
```

**Value:**

```
do { \
    pass &= test_impl<Simd##SIZE<Elt>, Elt>(); \
    cout << endl; \
} while (0)
```

**17.309.2 Function Documentation****17.309.2.1 check\_eq() [1/2]**

```
enable_if<is_integral<Element>::value, bool>::type check_eq (
    Element x,
    Element y )
```

**17.309.2.2 check\_eq() [2/2]**

```
enable_if<is_floating_point<Element>::value, bool>::type check_eq (
```

```

    Element x,
    Element y )

```

### 17.309.2.3 cmp()

```

bool cmp (
    vector< Element > out_scal,
    vector< Element > out_simd )

```

### 17.309.2.4 eval\_func\_on\_array() [1/3]

```

Ret eval_func_on_array (
    function< Ret()> f,
    array< T, 0 > & arr )

```

### 17.309.2.5 eval\_func\_on\_array() [2/3]

```

void eval_func_on_array (
    function< void(T, TArgs...)> f,
    array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr )

```

### 17.309.2.6 eval\_func\_on\_array() [3/3]

```

Ret eval_func_on_array (
    function< Ret(T, TArgs...)> f,
    array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr )

```

### 17.309.2.7 operator<<()

```

std::ostream& operator<< (
    std::ostream & o,
    const vector< E > & V )

```

### 17.309.2.8 test\_impl\_base() [1/2]

```

enable_if<is_floating_point<Element>::value, bool>::type test_impl_base ( )

```

### 17.309.2.9 test\_impl\_base() [2/2]

```

enable_if<is_integral<Element>::value, bool>::type test_impl_base ( )

```

### 17.309.2.10 test\_impl()

```

bool test_impl ( )

```

### 17.309.2.11 main()

```

int main (
    int argc,
    char * argv[] )

```

## 17.310 test-solve.C File Reference

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

### Functions

- template<typename Field , class RandIter >  
bool [check\\_solve](#) (const [Field](#) &F, size\_t m, RandIter &Rand, bool isParallel)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t iters, uint64\_t seed)
- int [main](#) (int argc, char \*\*argv)

### 17.310.1 Function Documentation

#### 17.310.1.1 [check\\_solve\(\)](#)

```
bool check_solve (
    const Field & F,
    size_t m,
    RandIter & Rand,
    bool isParallel )
```

#### 17.310.1.2 [run\\_with\\_field\(\)](#)

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t iters,
    uint64_t seed )
```

#### 17.310.1.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.311 test-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

## Data Structures

- class [Test<Elt>](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.311.1 Function Documentation

#### 17.311.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.312 test-utils.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include <givaro/givtimer.h>
#include <random>
#include <functional>
```

## Namespaces

- [FFLAS](#)
- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- uint64\_t [getSeed](#) ()
- template<typename Field >  
Field \* [chooseField](#) (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< int32\_t > \* [chooseField](#)< Givaro::ZRing< int32\_t > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< int64\_t > \* [chooseField](#)< Givaro::ZRing< int64\_t > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< float > \* [chooseField](#)< Givaro::ZRing< float > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- template<> Givaro::ZRing< double > \* [chooseField](#)< Givaro::ZRing< double > > (Givaro::Integer q, uint64\_t b, uint64\_t seed)

## 17.313 timer.h File Reference

```
#include <time.h>
#include <givaro/givtimer.h>
```

### Namespaces

- [FFLAS](#)

### Typedefs

- typedef Givaro::Timer [Timer](#)
- typedef Givaro::BaseTimer [BaseTimer](#)
- typedef Givaro::UserTimer [UserTimer](#)
- typedef Givaro::SysTimer [SysTimer](#)

## 17.314 utils.h File Reference

```
#include <algorithm>
#include <numeric>
#include <vector>
```

### Data Structures

- struct [StatsMatrix](#)

### Namespaces

- [FFLAS](#)

### Functions

- template<class It >  
double [computeDeviation](#) (It begin, It end)
- template<class Field >  
StatsMatrix [getStat](#) (const [Field](#) &F, const [index\\_t](#) \*row, const [index\\_t](#) \*col, typename [Field::ConstElement\\_ptr](#) val, uint64\_t rowdim, uint64\_t coldim, uint64\_t nnz)

## 17.315 winograd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <ctime>
```

### Macros

- #define [DOUBLE\\_TO\\_FLOAT\\_CROSSOVER](#) 0
- #define [GFOPS](#)(n, t) (2.0/t\*(double)n/1000.0\*(double)n/1000.0\*(double)n/1000.0)

## Typedefs

- typedef Givaro::Timer [TTimer](#)

## Functions

- template<class Field >  
bool [balanced](#) (const [Field](#) &)
- template<class T >  
bool [balanced](#) (const [Givaro::ModularBalanced](#)< T > &)
- int [main](#) ()

## 17.315.1 Macro Definition Documentation

### 17.315.1.1 DOUBLE\_TO\_FLOAT\_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

### 17.315.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t ) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

## 17.315.2 Typedef Documentation

### 17.315.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.315.3 Function Documentation

### 17.315.3.1 balanced() [1/2]

```
bool balanced (  
    const Field & )
```

### 17.315.3.2 balanced() [2/2]

```
bool balanced (  
    const Givaro::ModularBalanced< T > & )
```

### 17.315.3.3 main()

```
int main (  
    void )
```



# Index

- [\\_F](#)
  - [RNSIntegerMod< RNS >, 587](#)
- [\\_M](#)
  - [rns\\_double, 564](#)
  - [rns\\_double\\_extended, 577](#)
- [\\_MAX\\_SIZE\\_MATRICES](#)
  - [benchmark-checkers.C, 830](#)
- [\\_MMi](#)
  - [rns\\_double, 565](#)
  - [rns\\_double\\_extended, 577](#)
- [\\_Mi](#)
  - [rns\\_double, 564](#)
  - [rns\\_double\\_extended, 577](#)
- [\\_Mi\\_modp\\_rns](#)
  - [RNSIntegerMod< RNS >, 587](#)
- [\\_NR\\_TESTS](#)
  - [benchmark-checkers.C, 830](#)
- [\\_PLUQ](#)
  - [FFPACK, 384](#)
- [\\_RNSdelayed](#)
  - [RNSIntegerMod< RNS >, 588](#)
- [\\_TEST\\_ONE](#)
  - [test-simd.C, 1177](#)
- [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE](#)
  - [fflas\\_sparse.h, 990](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
  - [cblas.C, 858](#)
  - [clapack.C, 865](#)
  - [fblas.C, 902](#)
  - [lapack.C, 1092](#)
- [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#)
  - [fflas\\_pfgemm.inl, 980](#)
- [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)
  - [benchmark-charpoly-mp.C, 828](#)
- [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#)
  - [ffpack.h, 1025](#)
- [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#)
  - [ffpack.h, 1025](#)
- [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 905](#)
- [\\_\\_FFLASFFPACK\\_GAUSSJORDAN\\_BASECASE](#)
  - [ffpack\\_echelonforms.inl, 1055](#)
  - [test-echelon.C, 1132](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_BLAS](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#)
  - [cblas.C, 858](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CLAPACK](#)
  - [clapack.C, 866](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CXX11](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DGETRF](#)
  - [benchmark-dgetrf.C, 832](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DTRTRI](#)
  - [benchmark-dtrtri.C, 835](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INT128](#)
  - [fflas-ffpack/config.h, 878](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LAPACK](#)
  - [clapack.C, 866](#)
  - [fflas-ffpack/config.h, 879](#)
  - [lapack.C, 1092](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_MEMORY\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H](#)
  - [fflas-ffpack/config.h, 879](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H](#)
  - [fflas-ffpack/config.h, 879](#)

- \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_HAVE\_UNISTD\_H
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL
  - kaapi\_routines.inl, [1092](#)
- \_\_FFLASFFPACK\_LT\_OBJDIR
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_MINBLOCKCUTS
  - blockcuts.inl, [858](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET
  - benchmark-charpoly.C, [829](#)
  - benchmark-fadd-lvl2.C, [836](#)
  - benchmark-fdot.C, [836](#)
  - benchmark-fgemm-mp.C, [837](#)
  - benchmark-fgemm-rns.C, [838](#)
  - benchmark-fgemv-mp.C, [841](#)
  - benchmark-fgemv.C, [842](#)
  - benchmark-fgesv.C, [845](#)
  - benchmark-fsyrk.C, [846](#)
  - benchmark-fsytrf.C, [847](#)
  - benchmark-ftrsm-mp.C, [847](#)
  - benchmark-ftrsm.C, [848](#)
  - benchmark-ftrsv.C, [849](#)
  - benchmark-ftrtri.C, [849](#)
  - benchmark-pluq.C, [852](#)
  - benchmark-quasisep.C, [853](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_NAME
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_STRING
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_TARNAME
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_URL
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PACKAGE\_VERSION
  - fflas-ffpack/config.h, [880](#)
- \_\_FFLASFFPACK\_PLUQ\_THRESHOLD
  - fflas-ffpack-default-thresholds.h, [905](#)
  - test-echelon.C, [1132](#)
- \_\_FFLASFFPACK\_SEQPARTHRESHOLD
  - fflas\_pfgemm.inl, [980](#)
- \_\_FFLASFFPACK\_SEQUENTIAL
  - parallel.h, [1095](#)
  - test-echelon.C, [1132](#)
  - test-ftrmm.C, [1153](#)
  - test-ftrmv.C, [1154](#)
  - test-ftrsm.C, [1156](#)
  - test-ftrsv.C, [1159](#)
  - test-ftrtri.C, [1161](#)
  - test-lu.C, [1164](#)
  - test-rankprofiles.C, [1175](#)
- \_\_FFLASFFPACK\_SIZEOF\_CHAR
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_SIZEOF\_INT
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_SIZEOF\_SHORT
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_SIZEOF\_\_INT64\_T
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_STDC\_HEADERS
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_USE\_OPENMP
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_VERSION
  - fflas-ffpack/config.h, [881](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD
  - fflas-ffpack-default-thresholds.h, [904](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL
  - fflas-ffpack-default-thresholds.h, [904](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT
  - fflas-ffpack-default-thresholds.h, [905](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT
  - fflas-ffpack-default-thresholds.h, [904](#)
- \_\_FFLASFFPACK\_charpoly\_INL
  - ffpack\_charpoly.inl, [1050](#)
- \_\_FFLASFFPACK\_checker\_charpoly\_INL
  - checker\_charpoly.inl, [861](#)
- \_\_FFLASFFPACK\_checker\_det\_INL
  - checker\_det.inl, [861](#)
- \_\_FFLASFFPACK\_checker\_fgemm\_INL
  - checker\_fgemm.inl, [862](#)
- \_\_FFLASFFPACK\_checker\_ftrsm\_INL
  - checker\_ftrsm.inl, [862](#)
- \_\_FFLASFFPACK\_checker\_invert\_INL
  - checker\_invert.inl, [862](#)
- \_\_FFLASFFPACK\_checker\_pluq\_INL
  - checker\_pluq.inl, [863](#)
- \_\_FFLASFFPACK\_fadd\_INL
  - fflas\_fadd.inl, [925](#)
- \_\_FFLASFFPACK\_fassign\_INL
  - fflas\_fassign.inl, [926](#)
- \_\_FFLASFFPACK\_faxpy\_INL
  - fflas\_faxpy.inl, [927](#)
- \_\_FFLASFFPACK\_fdot\_INL
  - fflas\_fdot.inl, [928](#)
- \_\_FFLASFFPACK\_fflas\_blockcuts\_INL
  - blockcuts.inl, [858](#)
- \_\_FFLASFFPACK\_fflas\_bounds\_INL
  - fflas\_bounds.inl, [908](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL
  - fgemm\_classical.inl, [1075](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL
  - fgemm\_winograd.inl, [1078](#)

- \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL  
fflas\_level1.inl, [962](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL  
fflas\_level2.inl, [964](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL  
fflas\_level3.inl, [967](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL  
fflas\_helpers.inl, [947](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL  
fflas\_sparse.inl, [993](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL  
simd128.inl, [1117](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL  
simd128\_double.inl, [1117](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL  
simd128\_float.inl, [1118](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL  
simd128\_int16.inl, [1118](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL  
simd128\_int32.inl, [1119](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL  
simd128\_int64.inl, [1119](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL  
simd256.inl, [1120](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL  
simd256\_double.inl, [1120](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL  
simd256\_float.inl, [1121](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL  
simd256\_int16.inl, [1121](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL  
simd256\_int32.inl, [1122](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL  
simd256\_int64.inl, [1122](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_INL  
fflas\_freduce.inl, [937](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL  
fflas\_freduce\_mp.inl, [937](#)
- \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL  
fflas\_fsyr2k.inl, [941](#)
- \_\_FFLASFFPACK\_fflas\_fsyrk\_INL  
fflas\_fsyrk.inl, [943](#)
- \_\_FFLASFFPACK\_fflas\_fsyrk\_strassen\_INL  
fflas\_fsyrk\_strassen.inl, [944](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL  
igemm.inl, [1089](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL  
igemm\_kernels.inl, [1091](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL  
igemm\_tools.inl, [1091](#)
- \_\_FFLASFFPACK\_fflas\_pfgemm\_INL  
fflas\_pfgemm.inl, [980](#)
- \_\_FFLASFFPACK\_fflas\_pftsm\_INL  
fflas\_pftsm.inl, [981](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL  
csr\_hyb\_pspmm.inl, [886](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL  
csr\_hyb\_pspmv.inl, [886](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmmm\_INL  
csr\_hyb\_spmmm.inl, [887](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL  
csr\_hyb\_spmv.inl, [888](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL  
csr\_hyb\_utils.inl, [888](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL  
csr\_pspmm.inl, [889](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL  
csr\_pspmv.inl, [890](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmmm\_INL  
csr\_spmmm.inl, [891](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL  
csr\_spmv.inl, [892](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL  
ell\_pspmm.inl, [896](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL  
ell\_pspmv.inl, [896](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL  
ell\_simd\_pspmv.inl, [898](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL  
ell\_simd\_spmv.inl, [899](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL  
ell\_simd\_utils.inl, [899](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmmm\_INL  
ell\_spmmm.inl, [900](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL  
ell\_spmv.inl, [901](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL  
ell\_utils.inl, [902](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL  
hyb\_zo\_pspmm.inl, [1086](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL  
hyb\_zo\_pspmv.inl, [1086](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmmm\_INL  
hyb\_zo\_spmmm.inl, [1087](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL  
hyb\_zo\_spmv.inl, [1087](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL  
hyb\_zo\_utils.inl, [1088](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmmm\_INL  
coo\_spmmm.inl, [883](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL  
coo\_spmv.inl, [884](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL  
coo\_utils.inl, [884](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL  
sell\_pspmv.inl, [1115](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL  
sell\_spmv.inl, [1116](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL  
sell\_utils.inl, [1116](#)
- \_\_FFLASFFPACK\_ffpack\_INL  
ffpack.inl, [1027](#)
- \_\_FFLASFFPACK\_ffpack\_bruhatgen\_inl  
ffpack\_bruhatgen.inl, [1028](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL  
ffpack\_charpoly\_danilevski.inl, [1050](#)

- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL  
ffpack\_charpoly\_kgfast.inl, [1051](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL  
ffpack\_charpoly\_kgfastgeneralized.inl, [1052](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL  
ffpack\_charpoly\_kglu.inl, [1052](#)
- \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL  
ffpack\_echelonforms.inl, [1055](#)
- \_\_FFLASFFPACK\_ffpack\_fgesv\_INL  
ffpack\_fgesv.inl, [1055](#)
- \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL  
ffpack\_fgetrs.inl, [1056](#)
- \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL  
ffpack\_fsytrf.inl, [1058](#)
- \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL  
ffpack\_ftrssyr2k.inl, [1058](#)
- \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL  
ffpack\_ftrstr.inl, [1059](#)
- \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL  
ffpack\_ftrtr.inl, [1060](#)
- \_\_FFLASFFPACK\_ffpack\_invert\_INL  
ffpack\_invert.inl, [1066](#)
- \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL  
ffpack\_krylovelim.inl, [1066](#)
- \_\_FFLASFFPACK\_ffpack\_ludivine\_INL  
ffpack\_ludivine.inl, [1067](#)
- \_\_FFLASFFPACK\_ffpack\_minpoly\_INL  
ffpack\_minpoly.inl, [1069](#)
- \_\_FFLASFFPACK\_ffpack\_permutation\_INL  
ffpack\_permutation.inl, [1071](#)
- \_\_FFLASFFPACK\_ffpack\_pluq\_INL  
ffpack\_pluq.inl, [1072](#)
- \_\_FFLASFFPACK\_ffpack\_ppluq\_INL  
ffpack\_ppluq.inl, [1073](#)
- \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL  
ffpack\_rankprofiles.inl, [1075](#)
- \_\_FFLASFFPACK\_ffgemm\_INL  
fflas\_fgemm.inl, [930](#)
- \_\_FFLASFFPACK\_ffgemm\_bini\_INL  
schedule\_bini.inl, [1111](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_INL  
schedule\_winograd.inl, [1111](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_INL  
schedule\_winograd\_acc.inl, [1112](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_ip\_INL  
schedule\_winograd\_acc\_ip.inl, [1113](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_ip\_INL  
schedule\_winograd\_ip.inl, [1114](#)
- \_\_FFLASFFPACK\_ffgemv\_INL  
fflas\_ffgemv.inl, [931](#)
- \_\_FFLASFFPACK\_ffgemv\_mp\_INL  
fflas\_ffgemv\_mp.inl, [932](#)
- \_\_FFLASFFPACK\_ffger\_INL  
fflas\_ffger.inl, [933](#)
- \_\_FFLASFFPACK\_field\_rns\_INL  
rns.inl, [1110](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_INL  
rns-double.inl, [1108](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL  
rns-double-recint.inl, [1107](#)
- \_\_FFLASFFPACK\_freivalds\_INL  
fflas\_freivalds.inl, [938](#)
- \_\_FFLASFFPACK\_fscal\_INL  
fflas\_fscal.inl, [939](#)
- \_\_FFLASFFPACK\_fscal\_mp\_INL  
fflas\_fscal\_mp.inl, [940](#)
- \_\_FFLASFFPACK\_ftrmm\_INL  
fflas\_ftrmm.inl, [944](#)
- \_\_FFLASFFPACK\_ftrsm\_INL  
fflas\_ftrsm.inl, [945](#)
- \_\_FFLASFFPACK\_ftrsv\_INL  
fflas\_ftrsv.inl, [946](#)
- \_\_FFLASFFPACK\_simd512\_INL  
simd512.inl, [1123](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL  
simd512\_double.inl, [1123](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL  
simd512\_float.inl, [1124](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL  
simd512\_int32.inl, [1124](#)
- \_\_FFLAS\_L1\_INST\_C  
fflas\_L1\_inst.C, [949](#)
- \_\_FFLAS\_L2\_INST\_C  
fflas\_L2\_inst.C, [952](#)
- \_\_FFLAS\_L3\_INST\_C  
fflas\_L3\_inst.C, [957](#)
- \_\_FFLAS\_\_TRSM\_READONLY  
fflas\_L3\_inst\_implem.inl, [959](#)
- fflas\_level3.inl, [967](#)
- ffpack\_ppluq.inl, [1073](#)
- \_\_FFPACK\_FSYTRF\_BC\_CROUT  
benchmark-fsytrf.C, [847](#)
- \_\_FFPACK\_INST\_C  
ffpack\_inst.C, [1060](#)
- \_\_FFPACK\_charpoly\_mp\_INL  
ffpack\_charpoly\_mp.inl, [1053](#)
- \_\_FFPACK\_det\_mp\_INL  
ffpack\_det\_mp.inl, [1053](#)
- \_\_FFPACK\_ffgemm\_classical\_INL  
ffgemm\_classical\_mp.inl, [1077](#)
- \_\_FFPACK\_fger\_mp\_INL  
fflas\_fger\_mp.inl, [934](#)
- \_\_FFPACK\_ftrsm\_mp\_INL  
fflas\_ftrsm\_mp.inl, [946](#)
- \_\_FFPACK\_ludivine\_mp\_INL  
ffpack\_ludivine\_mp.inl, [1068](#)
- \_\_FFPACK\_pluq\_mp\_INL  
ffpack\_pluq\_mp.inl, [1073](#)
- \_\_LUDIVINE\_CUTOFF  
test-lu.C, [1164](#)
- \_\_has\_builtin  
bit\_manipulation.h, [856](#)
- \_\_alloc  
rns\_double\_elt, [567](#)
- rns\_double\_elt\_cstptr, [570](#)
- rns\_double\_elt\_ptr, [573](#)

- `_basis`
  - `rns_double`, [564](#)
  - `rns_double_extended`, [576](#)
- `_basisMax`
  - `rns_double`, [564](#)
  - `rns_double_extended`, [576](#)
- `_coo`
  - `SpMat< Field, flag >`, [805](#)
- `_coo16`
  - `CooMat< Field >`, [462](#)
- `_coo16_zo`
  - `CooMat< Field >`, [462](#)
- `_coo32`
  - `CooMat< Field >`, [462](#)
- `_coo32_zo`
  - `CooMat< Field >`, [462](#)
- `_coo64`
  - `CooMat< Field >`, [462](#)
- `_coo64_zo`
  - `CooMat< Field >`, [462](#)
- `_crt_in`
  - `rns_double`, [565](#)
  - `rns_double_extended`, [577](#)
- `_crt_out`
  - `rns_double`, [565](#)
  - `rns_double_extended`, [577](#)
- `_csr`
  - `SpMat< Field, flag >`, [805](#)
- `_csr16`
  - `CsrMat< Field >`, [464](#)
- `_csr16_zo`
  - `CsrMat< Field >`, [464](#)
- `_csr32`
  - `CsrMat< Field >`, [464](#)
- `_csr32_zo`
  - `CsrMat< Field >`, [465](#)
- `_csr64`
  - `CsrMat< Field >`, [464](#)
- `_csr64_zo`
  - `CsrMat< Field >`, [465](#)
- `_ell`
  - `SpMat< Field, flag >`, [805](#)
- `_ell16`
  - `EllMat< Field >`, [471](#)
- `_ell16_zo`
  - `EllMat< Field >`, [471](#)
- `_ell32`
  - `EllMat< Field >`, [471](#)
- `_ell32_zo`
  - `EllMat< Field >`, [472](#)
- `_ell64`
  - `EllMat< Field >`, [471](#)
- `_ell64_zo`
  - `EllMat< Field >`, [472](#)
- `_errorStream`
  - `Failure`, [473](#)
- `_field_rns`
  - `rns_double`, [564](#)
  - `rns_double_extended`, [577](#)
- `_iM_modp_rns`
  - `RNSIntegerMod< RNS >`, [587](#)
- `_ibeg`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, [494](#)
- `_iend`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, [494](#)
- `_invbasis`
  - `rns_double`, [564](#)
  - `rns_double_extended`, [576](#)
- `_jbeg`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, [494](#)
- `_jend`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, [494](#)
- `_ldm`
  - `rns_double`, [565](#)
  - `rns_double_extended`, [577](#)
- `_mi_sum`
  - `rns_double`, [565](#)
- `_mm`
  - `Test< Elt >`, [813](#)
- `_negbasis`
  - `rns_double`, [564](#)
  - `rns_double_extended`, [576](#)
- `_nn`
  - `Test< Elt >`, [813](#)
- `_p`
  - `RNSIntegerMod< RNS >`, [587](#)
- `_pbits`
  - `rns_double`, [565](#)
  - `rns_double_extended`, [577](#)
- `_ptr`
  - `rns_double_elt`, [566](#)
  - `rns_double_elt_cstptr`, [570](#)
  - `rns_double_elt_ptr`, [573](#)
- `_rns`
  - `RNSInteger< RNS >`, [581](#)
  - `RNSIntegerMod< RNS >`, [587](#)
- `_simd512_int64_INL`
  - `simd512_int64.inl`, [1125](#)
- `_size`
  - `rns_double`, [565](#)
  - `rns_double_extended`, [577](#)
- `_stride`
  - `rns_double_elt`, [567](#)
  - `rns_double_elt_cstptr`, [570](#)
  - `rns_double_elt_ptr`, [573](#)
- `_zero`
  - `ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >`, [596](#)
  - `ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >`, [598](#)
- `~CheckerImplem_Det`
  - `CheckerImplem_Det< Field >`, [441](#)
- `~CheckerImplem_PLUQ`
  - `CheckerImplem_PLUQ< Field >`, [446](#)
- `~CheckerImplem_charpoly`

- CheckerImplem\_charpoly< Field, Polynomial >, [439](#)
- CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, [441](#)
- ~CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, [443](#)
- ~CheckerImplem\_ftsrn
  - CheckerImplem\_ftsrn< Field >, [444](#)
- ~CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, [445](#)
- ~rns\_double\_elt
  - rns\_double\_elt, [566](#)
- 101-fgemm.C, [823](#)
  - main, [823](#)
- 2x2-fgemm.C, [823](#)
  - main, [823](#)
- 2x2-ftsrsv.C, [824](#)
  - main, [824](#)
- 2x2-pluq.C, [824](#)
  - main, [824](#)
- add
  - FFLAS::vectorised, [299](#)
  - FieldSimd< \_Field >, [477](#)
  - RNSIntegerMod< RNS >, [585](#)
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd256\_impl< true, true, false, 2 >, [679](#)
  - Simd256\_impl< true, true, false, 4 >, [696](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [717](#)
  - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [756](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- add\_r
  - FieldSimd< \_Field >, [477](#)
- addin
  - FieldSimd< \_Field >, [477](#)
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd256\_impl< true, true, false, 2 >, [679](#)
  - Simd256\_impl< true, true, false, 4 >, [696](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [717](#)
  - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
- Simd256\_impl< true, true, true, 8 >, [746](#)
- Simd512\_impl< true, false, true, 8 >, [756](#)
- Simd512\_impl< true, true, false, 8 >, [767](#)
- Simd512\_impl< true, true, true, 8 >, [776](#)
- addin\_r
  - FieldSimd< \_Field >, [477](#)
- addp
  - FFLAS::vectorised, [298](#)
- AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, [427](#)
  - value, [427](#)
- AlgoChooser< ModeT, ParSeq >, [427](#)
  - value, [427](#)
- align-allocator.h, [825](#)
- alignable
  - FFLAS, [201](#)
- alignable< Givaro::Integer \* >
  - FFLAS, [201](#)
- aligned\_allocator
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [612](#)
  - Simd128\_impl< true, true, false, 8 >, [622](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [685](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [725](#)
  - Simd256\_impl< true, true, true, 8 >, [743](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [761](#)
  - Simd512\_impl< true, true, true, 8 >, [772](#)
- aligned\_vector
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [612](#)
  - Simd128\_impl< true, true, false, 8 >, [622](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [685](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [725](#)
  - Simd256\_impl< true, true, true, 8 >, [743](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [773](#)
- alignment
  - FieldSimd< \_Field >, [480](#)
  - NoSimd< T >, [554](#)

- Simd128\_impl< true, true, false, 2 >, [610](#)
- Simd128\_impl< true, true, false, 4 >, [620](#)
- Simd128\_impl< true, true, false, 8 >, [631](#)
- Simd128\_impl< true, true, true, 2 >, [641](#)
- Simd128\_impl< true, true, true, 4 >, [650](#)
- Simd128\_impl< true, true, true, 8 >, [661](#)
- Simd256\_impl< true, false, true, 8 >, [671](#)
- Simd256\_impl< true, true, false, 2 >, [681](#)
- Simd256\_impl< true, true, false, 4 >, [701](#)
- Simd256\_impl< true, true, false, 8 >, [711](#)
- Simd256\_impl< true, true, true, 2 >, [721](#)
- Simd256\_impl< true, true, true, 4 >, [740](#)
- Simd256\_impl< true, true, true, 8 >, [750](#)
- Simd512\_impl< true, false, true, 8 >, [759](#)
- Simd512\_impl< true, true, false, 8 >, [770](#)
- Simd512\_impl< true, true, true, 8 >, [781](#)
- ALL< false, v... >, [428](#)
  - value, [428](#)
- ALL< true, v... >, [428](#)
  - value, [428](#)
- ALL< v >, [427](#)
- ALL<>, [428](#)
  - value, [428](#)
- Amax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- Amin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- applyP
  - FFPACK, [323](#), [324](#), [386](#)
- applyP\_block
  - FFPACK, [378](#)
- applyP\_modular\_double
  - ffpack.C, [1000](#)
  - ffpack\_c.h, [1036](#)
- ArbitraryPrecIntTag, [428](#)
- areEqual
  - RNSIntegerMod< RNS >, [586](#)
- AreEqual< X, X >, [429](#)
  - value, [429](#)
- AreEqual< X, Y >, [429](#)
  - value, [429](#)
- args-parser.h, [825](#)
  - ArgumentType, [826](#)
  - END\_OF\_ARGUMENTS, [826](#)
  - findArgument, [826](#)
  - getListArgs, [826](#)
  - printHelpMessage, [826](#)
  - TYPE\_BOOL, [825](#)
  - TYPE\_DOUBLE, [826](#)
  - TYPE\_INT, [826](#)
  - TYPE\_INTEGER, [826](#)
  - type\_integer, [826](#)
  - TYPE\_INTLIST, [826](#)
  - TYPE\_LONGLONG, [826](#)
  - TYPE\_NONE, [826](#)
  - TYPE\_STR, [826](#)
  - TYPE\_UINT64, [826](#)
- Argument, [429](#)
  - c, [430](#)
  - data, [430](#)
  - example, [430](#)
  - helpString, [430](#)
  - type, [430](#)
- ArgumentType
  - args-parser.h, [826](#)
- ArithProg
  - FFPACK::Protected, [421](#)
- arithprog.C, [827](#)
  - CUBE, [827](#)
  - GFOPS, [827](#)
  - main, [828](#)
  - TTimer, [827](#)
- assign
  - RNSInteger< RNS >, [581](#)
  - RNSIntegerMod< RNS >, [585](#)
- associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [430](#)
  - field, [431](#)
  - type, [431](#)
- associatedDelayedField< const Givaro::Modular< T, X > >, [431](#)
  - field, [431](#)
  - type, [431](#)
- associatedDelayedField< const Givaro::ModularBalanced< T > >, [431](#)
  - field, [432](#)
  - type, [432](#)
- associatedDelayedField< const Givaro::ZRing< T > >, [432](#)
  - field, [432](#)
  - type, [432](#)
- associatedDelayedField< Field >, [430](#)
  - field, [430](#)
  - type, [430](#)
- assume\_aligned
  - fflas\_sparse.h, [990](#)
- AtlasConj
  - config-blas.h, [868](#)
- Aunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [532](#)
- aut
  - HelperFlag, [503](#)
- Auto, [432](#)
- autotune/charpoly.C
  - CUBE, [859](#)
  - GFOPS, [859](#)
  - main, [860](#)
  - TTimer, [859](#)
- autotune/pluq.C
  - CUBE, [1102](#)
  - GFOPS, [1103](#)
  - main, [1103](#)
  - TTimer, [1103](#)

- averageCol
  - StatsMatrix, [807](#)
- averageColDifference
  - StatsMatrix, [807](#)
- averageRow
  - StatsMatrix, [806](#)
- averageRowDifference
  - StatsMatrix, [807](#)
- axpy
  - FieldSimd< \_Field >, [479](#)
- axpy\_r
  - FieldSimd< \_Field >, [479](#), [480](#)
- axpyin
  - FieldSimd< \_Field >, [479](#)
  - RNSIntegerMod< RNS >, [586](#)
- axpyin\_r
  - FieldSimd< \_Field >, [480](#)
- axpyp
  - FFLAS::vectorised, [299](#), [300](#)
  - FFLAS::vectorised::unswitch, [303](#), [304](#)
- balanced
  - FieldTraits< FFPACK::RNSInteger< T > >, [482](#)
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, [482](#)
  - FieldTraits< Field >, [481](#)
  - FieldTraits< Givaro::Modular< Element > >, [483](#)
  - FieldTraits< Givaro::ModularBalanced< Element > >, [483](#)
  - FieldTraits< Givaro::ZRing< double > >, [484](#)
  - FieldTraits< Givaro::ZRing< float > >, [484](#)
  - FieldTraits< Givaro::ZRing< Givaro::Integer > >, [485](#)
  - FieldTraits< Givaro::ZRing< int16\_t > >, [485](#)
  - FieldTraits< Givaro::ZRing< int32\_t > >, [486](#)
  - FieldTraits< Givaro::ZRing< int64\_t > >, [486](#)
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, [487](#)
  - FieldTraits< Givaro::ZRing< uint16\_t > >, [488](#)
  - FieldTraits< Givaro::ZRing< uint32\_t > >, [488](#)
  - FieldTraits< Givaro::ZRing< uint64\_t > >, [489](#)
  - winograd.C, [1183](#)
- BARRIER
  - parallel.h, [1095](#)
- BASECASE\_K
  - test-lu.C, [1164](#)
- BaseTimer
  - FFLAS, [78](#)
- BasisElement
  - rns\_double, [561](#)
  - rns\_double\_extended, [574](#)
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [583](#)
- begin
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - Info, [508](#), [510](#)
- BEGIN\_PARALLEL\_MAIN
  - parallel.h, [1096](#)
- Bench
  - Bench< Elt >, [434](#)
- Bench< Elt >, [432](#)
  - Bench, [434](#)
  - cardinality, [434](#)
  - doBenchs, [434](#)
  - Elt\_ptr, [433](#)
  - enable\_if\_no\_simd\_t, [434](#)
  - enable\_if\_simd128\_t, [434](#)
  - enable\_if\_simd256\_t, [434](#)
  - enable\_if\_simd512\_t, [434](#)
  - enable\_if\_t, [433](#)
  - F, [435](#)
  - Field, [433](#)
  - inplace, [435](#)
  - is\_same\_element, [433](#)
  - iters, [435](#)
  - m, [435](#)
  - n, [435](#)
  - Residu, [433](#)
  - run, [434](#)
- benchmark-charpoly-mp.C, [828](#)
  - \_\_FFLASFFPACK\_FORCE\_SEQ, [828](#)
  - main, [828](#)
- benchmark-charpoly.C, [828](#)
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, [829](#)
  - main, [829](#)
  - run\_with\_field, [829](#)
- benchmark-checkers.C, [829](#)
  - \_MAX\_SIZE\_MATRICES, [830](#)
  - \_NR\_TESTS, [830](#)
  - CUBE, [830](#)
  - ENABLE\_ALL\_CHECKINGS, [830](#)
  - main, [830](#)
- benchmark-dgemm.C, [830](#)
  - CBLAS\_GEMM, [831](#)
  - Floats, [831](#)
  - main, [831](#)
  - TTimer, [831](#)
- benchmark-dgetrf.C, [831](#)
  - \_\_FFLASFFPACK\_HAVE\_DGETRF, [832](#)
  - main, [832](#)
  - TTimer, [832](#)
- benchmark-dgetri.C, [832](#)
  - main, [833](#)
  - TTimer, [833](#)
- benchmark-dsytrf.C, [833](#)
  - EFFGFF, [833](#)
  - main, [834](#)
  - TTimer, [834](#)
- benchmark-dtrsm.C, [834](#)
  - main, [834](#)
  - TTimer, [834](#)
- benchmark-dtrtri.C, [835](#)
  - \_\_FFLASFFPACK\_HAVE\_DTRTRI, [835](#)
  - main, [835](#)
  - TTimer, [835](#)
- benchmark-fadd-lvl2.C, [835](#)

- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 836
- main, 836
- benchmark-fdot.C, 836
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 836
- main, 837
- run\_with\_field, 837
- benchmark-fgemm-mp.C, 837
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 837
- main, 838
- tmain, 837
- benchmark-fgemm-rns.C, 838
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 838
- ConstElement\_ptr, 839
- Element\_ptr, 839
- Field, 838
- GRAIN, 839
- main, 839
- PSeq, 839
- RNS, 838
- THREADS, 839
- THREED, 839
- THREEDA, 839
- THREEDIP, 839
- TWOD, 839
- TWODA, 839
- benchmark-fgemm.C, 840
- CLASSIC\_HYBRID, 840
- main, 840
- benchmark-fgemv-mp.C, 840
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 841
- write\_matrix, 841
- benchmark-fgemv.C, 841
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 842
- benchmark\_disp, 843
- benchmark\_in\_Field, 843
- benchmark\_with\_field, 844
- benchmark\_with\_timer, 843
- check\_result, 843
- fill\_value, 842
- genData, 842
- main, 844
- benchmark-fgesv.C, 844
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 845
- main, 845
- benchmark-fsyr2k.C, 845
- main, 845
- benchmark-fsyrk.C, 846
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 846
- main, 846
- benchmark-fsytrf.C, 846
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 847
- \_\_FFPACK\_FSYTRF\_BC\_CROUT, 847
- CUBE, 847
- main, 847
- benchmark-ftrsm-mp.C, 847
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 847
- main, 848
- benchmark-ftrsm.C, 848
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 848
- main, 848
- benchmark-ftrsv.C, 848
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 849
- main, 849
- benchmark-ftrtri.C, 849
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 849
- CUBE, 849
- main, 850
- benchmark-inverse.C, 850
- CUBE, 850
- main, 850
- benchmark-lqup-mp.C, 850
- main, 851
- benchmark-lqup.C, 851
- CUBE, 851
- main, 851
- benchmark-pluq.C, 852
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 852
- CUBE, 852
- Field, 852
- main, 853
- Rec\_Initialize, 853
- verification\_PLUQ, 852
- benchmark-quasisep.C, 853
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 853
- main, 854
- run\_with\_field, 854
- benchmark-storage-transpose.C, 854
- main, 854
- benchmark-wino.C, 855
- CUBE, 855
- launch\_wino, 855
- main, 855
- benchmark\_disp
- benchmark-fgemv.C, 843
- benchmark\_in\_Field
- benchmark-fgemv.C, 843
- benchmark\_with\_field
- benchmark-fgemv.C, 844
- benchmark\_with\_timer
- benchmark-fgemv.C, 843
- Bini, 435

- FFLAS::BLAS3, [206](#)
- bit\_manipulation.h, [855](#)
  - \_\_has\_builtin, [856](#)
  - clz, [856](#)
  - ctz, [856](#)
- bitsize
  - FFLAS, [149](#)
- bitsize< Givaro::ZRing< Givaro::Integer > >
  - FFLAS, [149](#)
- blas\_enum
  - config-blas.h, [867](#)
- blend
  - ScalFunctions< Element >, [594](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd256\_impl< true, true, false, 2 >, [679](#)
  - Simd256\_impl< true, true, false, 4 >, [696](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [717](#)
  - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [756](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- blendv
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [595](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd512\_impl< true, false, true, 8 >, [756](#)
- Block, [435](#)
- BlockCuts
  - FFLAS, [191](#), [193](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >
  - FFLAS, [193](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >
  - FFLAS, [192](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >
  - FFLAS, [193](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >
  - FFLAS, [192](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >
  - FFLAS, [192](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >
  - FFLAS, [193](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >
  - FFLAS, [192](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >
  - FFLAS, [192](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >
  - FFLAS, [193](#)
- BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >
  - FFLAS, [191](#)
- blockcuts.inl, [856](#)
  - \_\_FFLASFFPACK\_MINBLOCKCUTS, [858](#)
  - \_\_FFLASFFPACK\_fflas\_blockcuts\_INL, [858](#)
- blockindex
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [494](#)
- BlockingFactor
  - FFLAS::details, [220](#)
- BLOCKS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- BlockTransposeSIMD< Field, Simd, >, [435](#)
  - info, [436](#)
  - size, [436](#)
  - transpose, [436](#)
- Bmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [534](#)
- Bmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [534](#)
- Bruhat2EchelonPermutation
  - FFPACK, [366](#)
- build
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
- buildMatrix
  - FFPACK, [371](#)
- Bunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [532](#)
- c
  - Argument, [430](#)
- callLUdivine\_small< double >, [437](#)
  - operator(), [437](#)
- callLUdivine\_small< Element >, [437](#)
  - operator(), [437](#)
- callLUdivine\_small< float >, [438](#)
  - operator(), [438](#)
- cardinality
  - Bench< Elt >, [434](#)
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [584](#)
  - Test< Elt >, [812](#)
- cast.h, [858](#)
- category
  - FieldTraits< FFPACK::RNSInteger< T > >, [481](#)
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, [482](#)
  - FieldTraits< Field >, [481](#)

- FieldTraits< Givaro::Modular< Element > >, [483](#)
- FieldTraits< Givaro::ModularBalanced< Element > >, [483](#)
- FieldTraits< Givaro::ZRing< double > >, [484](#)
- FieldTraits< Givaro::ZRing< float > >, [484](#)
- FieldTraits< Givaro::ZRing< Givaro::Integer > >, [485](#)
- FieldTraits< Givaro::ZRing< int16\_t > >, [485](#)
- FieldTraits< Givaro::ZRing< int32\_t > >, [486](#)
- FieldTraits< Givaro::ZRing< int64\_t > >, [486](#)
- FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, [487](#)
- FieldTraits< Givaro::ZRing< uint16\_t > >, [487](#)
- FieldTraits< Givaro::ZRing< uint32\_t > >, [488](#)
- FieldTraits< Givaro::ZRing< uint64\_t > >, [488](#)
- cblas.C, [858](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [858](#)
  - \_\_FFLASFFPACK\_HAVE\_CBLAS, [858](#)
  - main, [859](#)
- CBLAS\_DIAG
  - config-blas.h, [868](#)
- cblas\_dsyrc
  - config-blas.h, [873](#)
- CBLAS\_ENUM\_DEFINED\_H
  - config-blas.h, [867](#)
- CBLAS\_EXTERNALS
  - config-blas.h, [867](#)
- CBLAS\_GEMM
  - benchmark-dgemm.C, [831](#)
- cblas\_imprsm
  - FFLAS, [137](#)
- CBLAS\_INT
  - config-blas.h, [867](#)
- CBLAS\_ORDER
  - config-blas.h, [867](#)
- CBLAS\_SIDE
  - config-blas.h, [868](#)
- CBLAS\_TRANSPOSE
  - config-blas.h, [867](#)
- CBLAS\_UPLO
  - config-blas.h, [868](#)
- CblasColMajor
  - config-blas.h, [867](#)
- CblasConjTrans
  - config-blas.h, [868](#)
- CblasLeft
  - config-blas.h, [868](#)
- CblasLower
  - config-blas.h, [868](#)
- CblasNonUnit
  - config-blas.h, [868](#)
- CblasNoTrans
  - config-blas.h, [868](#)
- CblasRight
  - config-blas.h, [868](#)
- CblasRowMajor
  - config-blas.h, [867](#)
- CblasTrans
  - config-blas.h, [868](#)
- CblasUnit
  - config-blas.h, [868](#)
- CblasUpper
  - config-blas.h, [868](#)
- ceil
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [595](#)
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd512\_impl< true, false, true, 8 >, [759](#)
- changeBS
  - ForStrategy1D< blocksize\_t, Cut, Param >, [492](#)
- changeCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- changeRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- characteristic
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [584](#)
- CharPoly
  - FFPACK, [342](#), [343](#), [372](#), [391](#), [392](#)
- charpoly.C, [859](#), [860](#)
- CharpolyFailed, [438](#)
- check
  - Checker\_Empty< Field >, [439](#)
  - CheckerImplem\_charpoly< Field, Polynomial >, [440](#)
  - CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, [441](#)
  - CheckerImplem\_Det< Field >, [442](#)
  - CheckerImplem\_fgemm< Field >, [443](#)
  - CheckerImplem\_ftsm< Field >, [444](#)
  - CheckerImplem\_invert< Field >, [445](#)
  - CheckerImplem\_PLUQ< Field >, [446](#)
- check1
  - regression-check.C, [1105](#)
- check2
  - regression-check.C, [1105](#)
- check3
  - regression-check.C, [1106](#)
- check4
  - regression-check.C, [1106](#)
- check\_computeS1S2
  - test-fsyrc.C, [1150](#)
- CHECK\_DEPENDENCIES
  - parallel.h, [1095](#)
- check\_eq
  - test-simd.C, [1178](#)
- check\_fdot
  - test-fdot.C, [1136](#)
- check\_fger
  - test-fger.C, [1142](#)
- check\_fsyrc2k
  - test-fsyrc2k.C, [1148](#)
- check\_fsyrc
  - test-fsyrc.C, [1149](#)
- check\_fsyrc\_bkdiag

- test-fsyrrk.C, 1150
- check\_fsyrrk\_diag
  - test-fsyrrk.C, 1150
- check\_ftrmm
  - test-ftrmm.C, 1153
- check\_ftrmv
  - test-ftrmv.C, 1154
- check\_ftrsm
  - test-ftrsm.C, 1156
- check\_ftrssyr2k
  - test-ftrssyr2k.C, 1157
- check\_ftrstr
  - test-ftrstr.C, 1158
- check\_ftrsv
  - test-ftrsv.C, 1160
- check\_ftrtri
  - test-ftrtri.C, 1161
- check\_minpoly
  - test-minpoly.C, 1169
- check\_MM
  - test-fgemm.C, 1138
- check\_MV
  - test-fgemv.C, 1140
- check\_result
  - benchmark-fgemv.C, 843
- check\_solve
  - test-solve.C, 1180
- checkA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 532
- checkB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 533
- CHECKER, 45
- Checker\_charpoly
  - FFPACK, 322
- checker\_charpoly.inl, 860
  - \_\_FFLASFFPACK\_checker\_charpoly\_INL, 861
- Checker\_Det
  - FFPACK, 322
- checker\_det.inl, 861
  - \_\_FFLASFFPACK\_checker\_det\_INL, 861
- Checker\_Empty
  - Checker\_Empty< Field >, 439
- Checker\_Empty< Field >, 438
  - check, 439
  - Checker\_Empty, 439
- checker\_empty.h, 861
- Checker\_fgemm
  - FFLAS, 76
- checker\_fgemm.inl, 861
  - \_\_FFLASFFPACK\_checker\_fgemm\_INL, 862
- Checker\_ftrsm
  - FFLAS, 76
- checker\_ftrsm.inl, 862
  - \_\_FFLASFFPACK\_checker\_ftrsm\_INL, 862
- Checker\_invert
  - FFPACK, 322
- checker\_invert.inl, 862
  - \_\_FFLASFFPACK\_checker\_invert\_INL, 862
- Checker\_PLUQ
  - FFPACK, 322
- checker\_pluq.inl, 863
  - \_\_FFLASFFPACK\_checker\_pluq\_INL, 863
- CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, 439
  - CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 440
- CheckerImplem\_charpoly< Field, Polynomial >, 439
  - ~CheckerImplem\_charpoly, 439
  - check, 440
  - CheckerImplem\_charpoly, 439
- CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 440
  - ~CheckerImplem\_charpoly, 441
  - check, 441
  - CheckerImplem\_charpoly, 440
  - Ring, 440
- CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, 441
- CheckerImplem\_Det< Field >, 441
  - ~CheckerImplem\_Det, 441
  - check, 442
  - CheckerImplem\_Det, 441
- CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, 442
- CheckerImplem\_fgemm< Field >, 442
  - ~CheckerImplem\_fgemm, 443
  - check, 443
  - CheckerImplem\_fgemm, 442
- CheckerImplem\_ftrsm
  - CheckerImplem\_ftrsm< Field >, 443, 444
- CheckerImplem\_ftrsm< Field >, 443
  - ~CheckerImplem\_ftrsm, 444
  - check, 444
  - CheckerImplem\_ftrsm, 443, 444
- CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, 444, 445
- CheckerImplem\_invert< Field >, 444
  - ~CheckerImplem\_invert, 445
  - check, 445
  - CheckerImplem\_invert, 444, 445
- CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, 445, 446
- CheckerImplem\_PLUQ< Field >, 445
  - ~CheckerImplem\_PLUQ, 446
  - check, 446
  - CheckerImplem\_PLUQ, 445, 446
- checkers.doxy, 863
- checkers\_fflas.h, 863
- checkers\_fflas.inl, 864
  - FFLASFFPACK\_checkers\_fflas\_inl\_H, 864
- checkers\_ffpack.h, 864
- checkers\_ffpack.inl, 865
  - FFLASFFPACK\_checkers\_ffpack\_inl\_H, 865

- checkingMessage
  - test-nullspace.C, [1170](#)
- checkMonotonicApplyP
  - test-permutations.C, [1171](#)
- checkOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [533](#)
- checkRPM
  - test-rpm.C, [1176](#)
- checkSymmetricRPM
  - test-rpm.C, [1176](#)
- checkZeroDimCharpoly
  - regression-check.C, [1106](#)
- checkZeroDimMinPoly
  - regression-check.C, [1106](#)
- chooseField
  - FFPACK, [416](#)
- chooseField< Givaro::ZRing< double > >
  - FFPACK, [417](#)
- chooseField< Givaro::ZRing< float > >
  - FFPACK, [416](#)
- chooseField< Givaro::ZRing< int32\_t > >
  - FFPACK, [416](#)
- chooseField< Givaro::ZRing< int64\_t > >
  - FFPACK, [416](#)
- chunk
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [794](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
- chunkSize
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [802](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- clapack.C, [865](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [865](#)
  - \_\_FFLASFFPACK\_HAVE\_CLAPACK, [866](#)
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, [866](#)
  - main, [866](#)
- Classic, [446](#)
- CLASSIC\_HYBRID
  - benchmark-fgemm.C, [840](#)
- clz
  - bit\_manipulation.h, [856](#)
- Cmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- Cmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- cmp
  - test-simd.C, [1179](#)
- cmp\_false
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [596](#)
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [599](#)
- cmp\_true
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [596](#)
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [599](#)
- col
  - Coo< Field >, [460](#)
  - Coo< ValT, IdxT >, [458](#), [461](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [784](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [792](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [795](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [797](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [802](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- colblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [494](#)
- colBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- coldim
  - StatsMatrix, [806](#)
- colnumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [494](#)
- ColRankProfileSubmatrix
  - FFPACK, [355](#), [397](#)
- ColRankProfileSubmatrix\_modular\_double
  - ffpack.C, [1014](#)
  - ffpack\_c.h, [1047](#)
- ColRankProfileSubmatrixIndices
  - FFPACK, [354](#), [396](#)
- ColRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [1013](#)
  - ffpack\_c.h, [1046](#)
- Column, [446](#)
- ColumnEchelonForm
  - FFPACK, [335](#), [336](#), [390](#)
- ColumnEchelonForm\_modular\_double
  - ffpack.C, [1003](#)
  - ffpack\_c.h, [1039](#)
- ColumnEchelonForm\_modular\_float
  - ffpack.C, [1004](#)
  - ffpack\_c.h, [1039](#)
- ColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [1005](#)
  - ffpack\_c.h, [1040](#)
- ColumnRankProfile
  - FFPACK, [351](#), [352](#), [396](#)
- ColumnRankProfile\_modular\_double
  - ffpack.C, [1012](#)
  - ffpack\_c.h, [1045](#)
- COMMA

- parallel.h, 1098
- CompactElement< double >, 447
  - type, 447
- CompactElement< Element >, 447
  - type, 447
- CompactElement< float >, 447
  - type, 447
- CompactElement< int16\_t >, 447
  - type, 448
- CompactElement< int32\_t >, 448
  - type, 448
- CompactElement< int64\_t >, 448
  - type, 448
- compatible\_data\_type< Field >, 448
  - value, 448
- compatible\_data\_type< Givaro::ZRing< double > >, 449
  - value, 449
- compatible\_data\_type< Givaro::ZRing< float > >, 449
  - value, 449
- compliant
  - NoSimd< T >, 554
  - Simd128\_impl< true, true, false, 2 >, 605
  - Simd128\_impl< true, true, false, 4 >, 615
  - Simd128\_impl< true, true, false, 8 >, 626
  - Simd128\_impl< true, true, true, 2 >, 633
  - Simd128\_impl< true, true, true, 4 >, 643
  - Simd128\_impl< true, true, true, 8 >, 653
  - Simd256\_impl< true, false, true, 8 >, 665
  - Simd256\_impl< true, true, false, 2 >, 677
  - Simd256\_impl< true, true, false, 4 >, 692
  - Simd256\_impl< true, true, false, 8 >, 707
  - Simd256\_impl< true, true, true, 2 >, 714
  - Simd256\_impl< true, true, true, 4 >, 726, 732
  - Simd256\_impl< true, true, true, 8 >, 743
  - Simd512\_impl< true, false, true, 8 >, 753
  - Simd512\_impl< true, true, false, 8 >, 765
  - Simd512\_impl< true, true, true, 8 >, 773
- Compose
  - Compose< H1, H2 >, 449, 450
- Compose< H1, H2 >, 449
  - Compose, 449, 450
  - first\_component, 450
  - operator<<, 450
  - second\_component, 450
- composePermutationsLLL
  - FFPACK, 381
  - ffpack.C, 1000
  - ffpack\_c.h, 1035
- composePermutationsLLM
  - FFPACK, 382
  - ffpack.C, 999
  - ffpack\_c.h, 1035
- composePermutationsMLM
  - FFPACK, 382
  - ffpack.C, 1000
  - ffpack\_c.h, 1035
- CompressRows
  - FFPACK::Protected, 422
- CompressRowsQA
  - FFPACK::Protected, 423
- CompressRowsQK
  - FFPACK::Protected, 423
- CompressToBlockBiDiagonal
  - FFPACK, 364
- computeDeviation
  - FFLAS, 163
- computeFactorClassic
  - FFLAS::Protected, 226
- ComputeRPermutation
  - FFPACK, 366, 369
- computeS1S2
  - FFLAS, 132
- config-blas.h, 866
  - AtlasConj, 868
  - blas\_enum, 867
  - CBLAS\_DIAG, 868
  - cblas\_dsyrk, 873
  - CBLAS\_ENUM\_DEFINED\_H, 867
  - CBLAS\_EXTERNALS, 867
  - CBLAS\_INT, 867
  - CBLAS\_ORDER, 867
  - CBLAS\_SIDE, 868
  - CBLAS\_TRANSPOSE, 867
  - CBLAS\_UPLO, 868
  - CblasColMajor, 867
  - CblasConjTrans, 868
  - CblasLeft, 868
  - CblasLower, 868
  - CblasNonUnit, 868
  - CblasNoTrans, 868
  - CblasRight, 868
  - CblasRowMajor, 867
  - CblasTrans, 868
  - CblasUnit, 868
  - CblasUpper, 868
  - dasum\_, 869
  - daxpy\_, 868
  - dcopy\_, 870
  - ddot\_, 869
  - dgemm\_, 873
  - dgemv\_, 869
  - dger\_, 870
  - dnrm2\_, 869
  - dscal\_, 871
  - dtrmm\_, 872
  - dtrsm\_, 871
  - idamax\_, 869
  - saxpy\_, 869
  - scopy\_, 871
  - sdot\_, 869
  - sgemm\_, 872
  - sgemv\_, 870
  - sger\_, 870
  - sscal\_, 871
  - strmm\_, 872

- strsm\_, 871
- config.h, 873, 877
  - HAVE\_BLAS, 874
  - HAVE\_CBLAS, 874
  - HAVE\_CXX11, 874
  - HAVE\_DLFCN\_H, 874
  - HAVE\_FLOAT\_H, 874
  - HAVE\_INT128, 874
  - HAVE\_INTPTR\_T, 874
  - HAVE\_INTTYPES\_H, 874
  - HAVE\_LAPACK, 875
  - HAVE\_LIMITS\_H, 875
  - HAVE\_LITTLE\_ENDIAN, 875
  - HAVE\_MEMORY\_H, 875
  - HAVE\_PTHREAD\_H, 875
  - HAVE\_STDDEF\_H, 875
  - HAVE\_STDINT\_H, 875
  - HAVE\_STDLIB\_H, 875
  - HAVE\_STRING\_H, 875
  - HAVE\_STRINGS\_H, 875
  - HAVE\_SYS\_STAT\_H, 875
  - HAVE\_SYS\_TIME\_H, 875
  - HAVE\_SYS\_TYPES\_H, 876
  - HAVE\_UNISTD\_H, 876
  - LT\_OBJDIR, 876
  - OPENBLAS\_NUM\_THREADS, 876
  - PACKAGE, 876
  - PACKAGE\_BUGREPORT, 876
  - PACKAGE\_NAME, 876
  - PACKAGE\_STRING, 876
  - PACKAGE\_TARNAME, 876
  - PACKAGE\_URL, 876
  - PACKAGE\_VERSION, 876
  - SIZEOF\_\_INT64\_T, 877
  - SIZEOF\_CHAR, 876
  - SIZEOF\_INT, 877
  - SIZEOF\_LONG, 877
  - SIZEOF\_LONG\_LONG, 877
  - SIZEOF\_SHORT, 877
  - STDC\_HEADERS, 877
  - USE\_OPENMP, 877
  - VERSION, 877
- CONST
  - fflas\_simd.h, 985
- ConstElement\_ptr
  - benchmark-fgemm-rns.C, 839
  - rns\_double, 561
  - rns\_double\_extended, 574
  - RNSInteger< RNS >, 579
  - RNSIntegerMod< RNS >, 583
- CONSTREFERENCE
  - parallel.h, 1096
- convert
  - rns\_double, 563, 564
  - rns\_double\_extended, 575, 576
  - RNSInteger< RNS >, 580
  - RNSIntegerMod< RNS >, 585
- convert\_transpose
  - rns\_double, 563
- ConvertTo< T >, 457
- COO
  - FFLAS, 82
- Coo
  - Coo< Field >, 459
  - Coo< ValT, IdxT >, 457, 458, 461
- coo
  - HelperFlag, 503
- Coo< Field >, 459
  - col, 460
  - Coo, 459
  - deleted, 460
  - operator=, 459
  - row, 460
  - val, 460
- Coo< ValT, IdxT >, 457, 460
  - col, 458, 461
  - Coo, 457, 458, 461
  - operator=, 458, 461
  - row, 458, 461
  - Self, 457, 460
  - val, 458, 461
- coo.h, 881
- coo\_spmv.inl, 882
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, 883
- coo\_spmv.inl, 883
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, 884
- coo\_utils.inl, 884
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL, 884
- COO\_ZO
  - FFLAS, 82
- CooMat< Field >, 462
  - \_coo16, 462
  - \_coo16\_zo, 462
  - \_coo32, 462
  - \_coo32\_zo, 462
  - \_coo64, 462
  - \_coo64\_zo, 462
- count\_nonconst\_lvalue\_reference< const T &, O... >, 463
  - n, 463
- count\_nonconst\_lvalue\_reference< T >, 463
- count\_nonconst\_lvalue\_reference< T &, O... >, 463
  - n, 463
- count\_nonconst\_lvalue\_reference< T, O... >, 463
  - n, 463
- count\_nonconst\_lvalue\_reference<>, 464
  - n, 464
- CROUT
  - ffpack\_pluq.inl, 1072
- CSC
  - FFLAS, 82
- CSC\_ZO
  - FFLAS, 82
- CSR

- FFLAS, [82](#)
- csr
  - HelperFlag, [503](#)
- csr.h, [884](#)
- CSR\_HYB
  - FFLAS, [82](#)
- csr\_hyb.h, [885](#)
- csr\_hyb\_pspmm.inl, [885](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL, [886](#)
- csr\_hyb\_pspmv.inl, [886](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL, [886](#)
- csr\_hyb\_spm.inl, [887](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spm\_INL, [887](#)
- csr\_hyb\_spmv.inl, [887](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, [888](#)
- csr\_hyb\_utils.inl, [888](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL, [888](#)
- csr\_pspmm.inl, [888](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL, [889](#)
- csr\_pspmv.inl, [889](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL, [890](#)
- csr\_spm.inl, [890](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm\_INL, [891](#)
- csr\_spmv.inl, [891](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL, [892](#)
- csr\_utils.inl, [892](#)
- CSR\_ZO
  - FFLAS, [82](#)
- CsrMat< Field >, [464](#)
  - \_csr16, [464](#)
  - \_csr16\_zo, [464](#)
  - \_csr32, [464](#)
  - \_csr32\_zo, [465](#)
  - \_csr64, [464](#)
  - \_csr64\_zo, [465](#)
- cst
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [791](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
- ctz
  - bit\_manipulation.h, [856](#)
- CUBE
  - arithprog.C, [827](#)
  - autotune/charpoly.C, [859](#)
  - autotune/pluq.C, [1102](#)
  - benchmark-checkers.C, [830](#)
  - benchmark-fsytrf.C, [847](#)
  - benchmark-ftrtri.C, [849](#)
  - benchmark-inverse.C, [850](#)
  - benchmark-lqup.C, [851](#)
  - benchmark-pluq.C, [852](#)
  - benchmark-wino.C, [855](#)
  - fsyrk.C, [1082](#)
  - fsytrf.C, [1083](#)
  - cuda.C, [893](#)
  - main, [893](#)
  - current
    - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
    - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
  - Parallel< C, P >, [555](#)
  - cyclic\_shift\_col
    - FFPACK, [383](#), [386](#)
  - cyclic\_shift\_col\_modular\_double
    - ffpack.C, [1000](#)
    - ffpack\_c.h, [1035](#)
  - cyclic\_shift\_mathPerm
    - FFPACK, [382](#)
    - ffpack.C, [1000](#)
    - ffpack\_c.h, [1035](#)
  - cyclic\_shift\_row
    - FFPACK, [383](#), [386](#)
  - cyclic\_shift\_row\_col
    - FFPACK, [382](#), [386](#)
  - cyclic\_shift\_row\_modular\_double
    - ffpack.C, [1000](#)
    - ffpack\_c.h, [1035](#)
- Danilevski
  - FFPACK, [371](#)
  - FFPACK::Protected, [421](#)
- dasum\_
  - config-blas.h, [869](#)
- dat
  - Sparse< \_Field, SparseMatrix\_t::COO >, [784](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [792](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [795](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [797](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [800](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [802](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- data
  - Argument, [430](#)
- daxpy\_
  - config-blas.h, [868](#)
- dcopy\_
  - config-blas.h, [870](#)

- ddot\_
  - config-blas.h, [869](#)
- debug.h, [893](#)
  - FFLASFFPACK\_abort, [894](#)
  - FFLASFFPACK\_check, [894](#)
- DeCompressRows
  - FFPACK::Protected, [423](#)
- DeCompressRowsQA
  - FFPACK::Protected, [424](#)
- DeCompressRowsQK
  - FFPACK::Protected, [423](#)
- DefaultBoundedTag, [465](#)
- DefaultTag, [465](#)
- delayed
  - RNSIntegerMod< RNS >, [583](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [784](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [791](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [794](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [799](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
- DelayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [531](#)
- delayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- DelayedField\_t
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [531](#)
- DelayedTag, [465](#)
- deleted
  - Coo< Field >, [460](#)
- DENSE\_THRESHOLD
  - fflas\_sparse.h, [991](#)
- denseCols
  - StatsMatrix, [808](#)
- denseRows
  - StatsMatrix, [808](#)
- Det
  - FFPACK, [347](#), [372](#), [394](#)
- det.C, [894](#)
  - main, [894](#)
- Det\_modular\_double
  - ffpack.C, [1011](#)
  - ffpack\_c.h, [1044](#)
- deviationCol
  - StatsMatrix, [807](#)
- deviationColDifference
  - StatsMatrix, [807](#)
- deviationRow
  - StatsMatrix, [806](#)
- deviationRowDifference
  - StatsMatrix, [807](#)
- DFElt
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [531](#)
- dgemm\_
  - config-blas.h, [873](#)
  - fblas.C, [902](#)
- dgemv\_
  - config-blas.h, [869](#)
- dger\_
  - config-blas.h, [870](#)
- digits
  - limits< char >, [518](#)
  - limits< double >, [519](#)
  - limits< float >, [520](#)
  - limits< int >, [521](#)
  - limits< long >, [522](#)
  - limits< long long >, [523](#)
  - limits< short int >, [525](#)
  - limits< signed char >, [525](#)
  - limits< unsigned char >, [526](#)
  - limits< unsigned int >, [527](#)
  - limits< unsigned long >, [527](#)
  - limits< unsigned long long >, [528](#)
  - limits< unsigned short int >, [529](#)
- div
  - ScalFunctions< Element >, [592](#)
  - Simd256\_impl< true, false, true, 8 >, [669](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
- DivideAndConquer, [465](#)
- dnrm2\_
  - config-blas.h, [869](#)
- DNS\_BIN\_VER
  - read\_sparse.h, [1105](#)
- doApplyS
  - FFPACK, [378](#)
- doApplyT
  - FFPACK, [380](#)
- doBenchs
  - Bench< Elt >, [434](#)
- doTests
  - Test< Elt >, [812](#)
- DotProdBoundClassic
  - FFLAS::Protected, [227](#)
- DOUBLE\_TO\_FLOAT\_CROSSOVER
  - fflas.h, [907](#)
  - winograd.C, [1183](#)
- dscal\_
  - config-blas.h, [871](#)
- dtrmm\_
  - config-blas.h, [872](#)
- dtrsm\_
  - config-blas.h, [871](#)
- DynamicPeeling
  - FFLAS::Protected, [228](#)

- DynamicPeeling2
  - FFLAS::Protected, [229](#)
- EFFGFF
  - benchmark-dsytrf.C, [833](#)
- Element
  - FieldSimd< \_Field >, [475](#)
  - readMyMachineType< Field, mpz\_t >, [559](#)
  - readMyMachineType< Field, T >, [558](#)
  - rns\_double, [561](#)
  - rns\_double\_extended, [574](#)
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [583](#)
  - TestOneMethod< Simd >, [814](#)
- Element\_ptr
  - benchmark-fgemm-rns.C, [839](#)
  - readMyMachineType< Field, mpz\_t >, [559](#)
  - readMyMachineType< Field, T >, [558](#)
  - rns\_double, [561](#)
  - rns\_double\_extended, [574](#)
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [583](#)
- ElementTraits< double >, [466](#)
  - value, [466](#)
- ElementTraits< Element >, [466](#)
  - value, [466](#)
- ElementTraits< FFPACK::rns\_double\_elt >, [466](#)
  - value, [466](#)
- ElementTraits< float >, [467](#)
  - value, [467](#)
- ElementTraits< Givaro::Integer >, [467](#)
  - value, [467](#)
- ElementTraits< int16\_t >, [467](#)
  - value, [467](#)
- ElementTraits< int32\_t >, [468](#)
  - value, [468](#)
- ElementTraits< int64\_t >, [468](#)
  - value, [468](#)
- ElementTraits< int8\_t >, [468](#)
  - value, [468](#)
- ElementTraits< Reclnt::rint< K > >, [469](#)
  - value, [469](#)
- ElementTraits< Reclnt::rmint< K, MG > >, [469](#)
  - value, [469](#)
- ElementTraits< Reclnt::ruint< K > >, [469](#)
  - value, [469](#)
- ElementTraits< uint16\_t >, [470](#)
  - value, [470](#)
- ElementTraits< uint32\_t >, [470](#)
  - value, [470](#)
- ElementTraits< uint64\_t >, [470](#)
  - value, [470](#)
- ElementTraits< uint8\_t >, [471](#)
  - value, [471](#)
- ELL
  - FFLAS, [82](#)
- ell
  - HelperFlag, [503](#)
- ell.h, [894](#)
- ell\_pspmm.inl, [895](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL, [896](#)
- ell\_pspmv.inl, [896](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL, [896](#)
- ELL\_simd
  - FFLAS, [82](#)
- ell\_simd.h, [897](#)
- ell\_simd\_pspmv.inl, [897](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL, [898](#)
- ell\_simd\_spmv.inl, [898](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL, [899](#)
- ell\_simd\_utils.inl, [899](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL, [899](#)
- ELL\_simd\_ZO
  - FFLAS, [82](#)
- ell\_spm.inl, [899](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm\_INL, [900](#)
- ell\_spmv.inl, [900](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL, [901](#)
- ell\_utils.inl, [901](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL, [902](#)
- ELL\_ZO
  - FFLAS, [82](#)
- EllMat< Field >, [471](#)
  - \_ell16, [471](#)
  - \_ell16\_zo, [471](#)
  - \_ell32, [471](#)
  - \_ell32\_zo, [472](#)
  - \_ell64, [471](#)
  - \_ell64\_zo, [472](#)
- Elt\_ptr
  - Bench< Elt >, [433](#)
  - Test< Elt >, [811](#)
- ENABLE\_ALL\_CHECKINGS
  - benchmark-checkers.C, [830](#)
  - ffpack\_ftrtr.inl, [1060](#)
  - test-fdot.C, [1136](#)
  - test-fgemm-check.C, [1137](#)
  - test-fsyr2k.C, [1148](#)
  - test-fsyrk.C, [1149](#)
  - test-ftrmv.C, [1154](#)
  - test-ftrsm-check.C, [1155](#)
  - test-ftrsm.C, [1156](#)
  - test-ftrssyr2k.C, [1157](#)
  - test-ftrstr.C, [1158](#)
  - test-ftrsv.C, [1159](#)
  - test-ftrtri.C, [1161](#)
  - test-invert-check.C, [1162](#)
  - test-pluq-check.C, [1173](#)
- ENABLE\_CHECKER\_charpoly

- test-charpoly-check.C, [1127](#)
- ENABLE\_CHECKER\_Det
  - test-det-check.C, [1130](#)
- ENABLE\_CHECKER\_fgemmm
  - test-fgemmm.C, [1138](#)
- enable\_if\_no\_simd\_t
  - Bench< Elt >, [434](#)
  - Test< Elt >, [811](#)
- enable\_if\_simd128\_t
  - Bench< Elt >, [434](#)
  - Test< Elt >, [811](#)
- enable\_if\_simd256\_t
  - Bench< Elt >, [434](#)
  - Test< Elt >, [811](#)
- enable\_if\_simd512\_t
  - Bench< Elt >, [434](#)
  - Test< Elt >, [812](#)
- enable\_if\_t
  - Bench< Elt >, [433](#)
  - Test< Elt >, [811](#)
  - TestOneMethod< Simd >, [814](#)
- end
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
- END\_OF\_ARGUMENTS
  - args-parser.h, [826](#)
- END\_PARALLEL\_MAIN
  - parallel.h, [1096](#)
- eq
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [609](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [629](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [669](#)
  - Simd256\_impl< true, true, false, 2 >, [681](#)
  - Simd256\_impl< true, true, false, 4 >, [699](#)
  - Simd256\_impl< true, true, false, 8 >, [710](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [731](#), [738](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- eval\_func\_on\_array
  - test-simd.C, [1179](#)
- evaluate\_scalar\_method
  - TestOneMethod< Simd >, [814](#)
- evaluate\_simd\_method
  - TestOneMethod< Simd >, [815](#)
- example
  - Argument, [430](#)
- examples/charpoly.C
  - main, [860](#)
- examples/pluq.C
  - main, [1103](#)
- ExpandBlockBiDiagonalToBruhat
  - FFPACK, [365](#)
- expandLCRE
  - FFPACK, [369](#)
- F
  - Bench< Elt >, [435](#)
  - Test< Elt >, [812](#)
- fadd
  - FFLAS, [83–85](#), [87](#), [175](#), [183](#), [184](#)
  - FFLAS::details, [213](#), [214](#)
- fadd\_1\_modular\_double
  - fflas\_c.h, [915](#)
  - fflas\_lvl1.C, [971](#)
- fadd\_2\_modular\_double
  - fflas\_c.h, [919](#)
  - fflas\_lvl2.C, [975](#)
- faddin
  - FFLAS, [83](#), [87](#), [175](#), [185](#)
- faddin\_1\_modular\_double
  - fflas\_c.h, [915](#)
  - fflas\_lvl1.C, [971](#)
- faddin\_2\_modular\_double
  - fflas\_c.h, [919](#)
  - fflas\_lvl2.C, [976](#)
- Failure, [472](#)
  - \_errorStream, [473](#)
  - Failure, [472](#)
  - operator(), [472](#), [473](#)
  - print, [473](#)
  - setErrorStream, [473](#)
- failure
  - FFPACK, [400](#)
- FailureCharpolyCheck, [474](#)
- FailureDetCheck, [474](#)
- FailureFgemmmCheck, [474](#)
- FailureInvertCheck, [474](#)
- FailurePLUQCheck, [474](#)
- FailureTrsmCheck, [474](#)
- fassign
  - FFLAS, [88–90](#), [171](#), [176](#)
- fassign\_1\_modular\_double
  - fflas\_c.h, [914](#)
  - fflas\_lvl1.C, [969](#)
- fassign\_2\_modular\_double
  - fflas\_c.h, [916](#)
  - fflas\_lvl2.C, [973](#)
- faxpby
  - FFLAS, [142](#), [148](#)
- faxpy
  - FFLAS, [91](#), [173](#), [181](#)
  - FFLAS::details, [215](#)
- faxpy\_1\_modular\_double
  - fflas\_c.h, [914](#)
  - fflas\_lvl1.C, [970](#)
- faxpy\_2\_modular\_double
  - fflas\_c.h, [918](#)
  - fflas\_lvl2.C, [975](#)
- fblas.C, [902](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [902](#)

- dgemm\_, 902
  - main, 903
- fconvert
  - FFLAS, 139, 146, 169
- fconvert\_rns
  - FFLAS, 166, 167
- fconvert\_trans\_rns
  - FFLAS, 166
- fdot
  - FFLAS, 92–94, 142, 173, 194
- fdot\_1\_modular\_double
  - fflas\_c.h, 915
  - fflas\_lvl1.C, 970
- fequal
  - FFLAS, 141, 144, 171, 177
- fequal\_1\_modular\_double
  - fflas\_c.h, 914
  - fflas\_lvl1.C, 969
- fequal\_2\_modular\_double
  - fflas\_c.h, 916
  - fflas\_lvl2.C, 973
- FFLAS, 45, 49
  - alignable, 201
  - alignable< Givaro::Integer \* >, 201
  - BaseTimer, 78
  - bitsize, 149
  - bitsize< Givaro::ZRing< Givaro::Integer > >, 149
  - BlockCuts, 191, 193
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >, 193
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >, 192
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >, 193
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >, 192
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >, 192
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >, 193
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >, 192
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >, 192
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >, 193
  - BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >, 191
  - cblas\_imptrsm, 137
  - Checker\_fgemm, 76
  - Checker\_ftrsm, 76
  - computeDeviation, 163
  - computeS1S2, 132
  - COO, 82
  - COO\_ZO, 82
  - CSC, 82
  - CSC\_ZO, 82
  - CSR, 82
  - CSR\_HYB, 82
  - CSR\_ZO, 82
  - ELL, 82
  - ELL\_simd, 82
  - ELL\_simd\_ZO, 82
  - ELL\_ZO, 82
  - fadd, 83–85, 87, 175, 183, 184
  - faddin, 83, 87, 175, 185
  - fassign, 88–90, 171, 176
  - faxpby, 142, 148
  - faxpy, 91, 173, 181
  - fconvert, 139, 146, 169
  - fconvert\_rns, 166, 167
  - fconvert\_trans\_rns, 166
  - fdot, 92–94, 142, 173, 194
  - fequal, 141, 144, 171, 177
  - FFLAS\_BASE, 81
  - fflas\_delete, 164, 165, 202
  - FFLAS\_DIAG, 80
  - FFLAS\_FORMAT, 82
  - fflas\_new, 165, 166, 201
  - FFLAS\_ORDER, 79
  - FFLAS\_SIDE, 81
  - FFLAS\_TRANSPOSE, 80
  - FFLAS\_UPLO, 80
  - FflasAuto, 82
  - FflasBinary, 82
  - FflasColMajor, 80
  - FflasDense, 82
  - FflasDouble, 81
  - FflasFloat, 81
  - FflasGeneric, 81
  - FflasLeft, 81
  - FflasLeftTri, 80
  - FflasLower, 80
  - FflasMaple, 82
  - FflasMath, 82
  - FflasNonUnit, 80
  - FflasNoTrans, 80
  - FflasRight, 81
  - FflasRightTri, 80
  - FflasRowMajor, 80
  - FflasSageMath, 82
  - FflasSMS, 82
  - FflasTrans, 80
  - FflasUnit, 80
  - FflasUpper, 80
  - fgemm, 94–101, 153, 189, 190
  - fgemv, 104–109, 185, 197
  - fger, 109–113, 186
  - fidentity, 145, 146, 178
  - finit, 115, 117, 138, 146, 168, 179
  - finit\_rns, 165, 167
  - finit\_trans\_rns, 165
  - fiszero, 141, 145, 170, 177
  - fmove, 148, 182
  - fneg, 139, 147, 169, 180
  - fnegin, 139, 147, 169, 179

ForceCheck\_fgemm, 76  
 ForceCheck\_ftrsm, 76  
 frand, 140, 144  
 freduce, 113–116, 118, 167, 168, 178, 179  
 freduce\_constoverride, 114, 117  
 freivalds, 118  
 fscal, 119–123, 172, 181  
 fscaln, 119–123, 172, 180  
 fspmm, 164  
 fspmv, 162, 163  
 fsquare, 102, 103, 191  
 fsub, 84, 86, 175, 183  
 fsubin, 84, 87, 184  
 fswap, 142, 174  
 fsyr2k, 124  
 fsyrk, 125–130, 132, 133  
 fsyrk\_strassen, 133, 151  
 ftrmm, 133, 134, 188  
 ftrmv, 150  
 ftrsm, 135, 136, 150, 154, 187  
 ftrsv, 137, 186  
 fzero, 140, 143, 170, 176  
 getArgumentValue, 198  
 getDataType, 160, 161  
 getSeed, 203  
 getStat, 163  
 getTLBSize, 202  
 has\_equal, 77  
 has\_minus, 77  
 has\_minus\_eq, 78  
 has\_mul, 78  
 has\_mul\_eq, 78  
 has\_plus, 77  
 has\_plus\_eq, 77  
 HYB\_ZO, 82  
 igemm\_, 138  
 InfNorm, 82  
 max3, 83  
 max4, 83  
 maxCardinality, 164  
 maxCardinality< Givaro::Modular< int32\_t > >, 164  
 maxCardinality< Givaro::Modular< int64\_t > >, 164  
 min3, 82  
 min4, 83  
 minCardinality, 164  
 MKLSparseMatrixFormat, 77  
 mone, 81  
 NoSimdSparseMatrix, 77  
 NotMKLSparseMatrixFormat, 77  
 NotZOSparseMatrix, 76  
 number\_kind, 81  
 one, 81  
 operator<=, 160  
 other, 81  
 parseArguments, 198  
 pfadd, 84

pfaddin, 85  
 pfgemm, 151, 194–196  
 pfgemm\_1D\_rec, 152  
 pfgemm\_2D\_rec, 152  
 pfgemm\_3D\_rec, 152  
 pfgemm\_3D\_rec2, 153  
 pfrand, 194  
 pfreduce, 116  
 pfsb, 85  
 pfsbin, 85  
 pfzero, 194  
 preamble, 199  
 prefetch, 202  
 queryCacheSizes, 202  
 queryL1CacheSize, 202  
 queryTopLevelCacheSize, 202  
 readDnsFormat, 161  
 readMachineType, 161  
 ReadMatrix, 199, 200  
 readSmsFormat, 160  
 readSprFormat, 160  
 SELL, 82  
 SELL\_ZO, 82  
 SimdSparseMatrix, 76  
 sparse\_delete, 154, 155, 157–159, 162  
 sparse\_init, 155–160, 162, 163  
 sparse\_print, 156, 159, 162  
 SparseMatrix\_t, 81  
 SysTimer, 78  
 Timer, 78  
 UserTimer, 78  
 writeCommandString, 198  
 writeDnsFormat, 161  
 WriteMatrix, 198, 200  
 WritePermutation, 200  
 zero, 81  
 ZOSparseMatrix, 76  
 fflas-101\_1.C, 903  
   main, 903  
 fflas-101\_3.C, 903  
   main, 903  
 FFLAS-FFPACK, 46  
 FFLAS-FFPACK fields, 46  
 fflas-ffpack-config.h, 904  
   GCC\_VERSION, 904  
 fflas-ffpack-default-thresholds.h, 904  
   \_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_CHARPOLY\_Danilevskii\_LUKrylov\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_CHARPOLY\_LUKrylov\_ArithProg\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_FSYRK\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_FSYTRF\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_FTRTRI\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, 905  
   \_\_FFLASFFPACK\_WINOTHRESHOLD, 904  
   \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL, 904

- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT, 905
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT, 904
- fflas-ffpack-thresholds.h, 905
- fflas-ffpack.doxy, 905
- fflas-ffpack.h, 905
- fflas-ffpack/config.h
  - \_\_FFLASFFPACK\_HAVE\_BLAS, 878
  - \_\_FFLASFFPACK\_HAVE\_CBLAS, 878
  - \_\_FFLASFFPACK\_HAVE\_CXX11, 878
  - \_\_FFLASFFPACK\_HAVE\_DLFCN\_H, 878
  - \_\_FFLASFFPACK\_HAVE\_FLOAT\_H, 878
  - \_\_FFLASFFPACK\_HAVE\_INT128, 878
  - \_\_FFLASFFPACK\_HAVE\_INTPTR\_T, 879
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, 879
  - \_\_FFLASFFPACK\_HAVE\_LIMITS\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_LITTLE\_ENDIAN, 879
  - \_\_FFLASFFPACK\_HAVE\_MEMORY\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_PTHREAD\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_STDDEF\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_STDINT\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_STDLIB\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_STRINGS\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_STRING\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H, 879
  - \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H, 880
  - \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H, 880
  - \_\_FFLASFFPACK\_HAVE\_UNISTD\_H, 880
  - \_\_FFLASFFPACK\_LT\_OBJDIR, 880
  - \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS, 880
  - \_\_FFLASFFPACK\_PACKAGE, 880
  - \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT, 880
  - \_\_FFLASFFPACK\_PACKAGE\_NAME, 880
  - \_\_FFLASFFPACK\_PACKAGE\_STRING, 880
  - \_\_FFLASFFPACK\_PACKAGE\_TARNAME, 880
  - \_\_FFLASFFPACK\_PACKAGE\_URL, 880
  - \_\_FFLASFFPACK\_PACKAGE\_VERSION, 880
  - \_\_FFLASFFPACK\_SIZEOF\_CHAR, 881
  - \_\_FFLASFFPACK\_SIZEOF\_INT, 881
  - \_\_FFLASFFPACK\_SIZEOF\_LONG, 881
  - \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG, 881
  - \_\_FFLASFFPACK\_SIZEOF\_SHORT, 881
  - \_\_FFLASFFPACK\_SIZEOF\_\_INT64\_T, 881
  - \_\_FFLASFFPACK\_STDC\_HEADERS, 881
  - \_\_FFLASFFPACK\_USE\_OPENMP, 881
  - \_\_FFLASFFPACK\_VERSION, 881
- fflas.doxy, 906
- fflas.h, 906
  - DOUBLE\_TO\_FLOAT\_CROSSOVER, 907
  - WINOTHRESHOLD, 907
- FFLAS::ftranspose\_impl, 203
  - nonsquare\_inplace\_v1, 204
  - nonsquare\_inplace\_v2, 204
  - not\_inplace, 203
  - square\_inplace, 203
- FFLAS::BLAS3, 204
  - Bini, 206
  - Winograd, 206
  - Winograd\_L\_S, 210
  - Winograd\_LR\_S, 210
  - Winograd\_R\_S, 210
  - WinogradAcc\_2\_24, 208
  - WinogradAcc\_2\_27, 208
  - WinogradAcc\_3\_21, 207
  - WinogradAcc\_3\_23, 207
  - WinogradAcc\_L\_S, 209
  - WinogradAcc\_LR, 208
  - WinogradAcc\_R\_S, 209
  - WinoPar, 206
- FFLAS::csr\_hyb\_details, 211
- FFLAS::CuttingStrategy, 211
  - RNSModulus, 211
- FFLAS::details, 212
  - BlockingFactor, 220
  - fadd, 213, 214
  - faxpy, 215
  - freduce, 215, 216
  - fscale, 216, 217
  - fscalein, 216, 217
  - gebp, 220
  - igebb11, 219
  - igebb14, 218
  - igebb21, 218
  - igebb24, 217
  - igebb41, 218
  - igebb44, 217
  - igebp, 219
  - pack\_lhs, 219
  - pack\_rhs, 220
- FFLAS::details\_spmv, 221
- FFLAS::ElementCategories, 221
- FFLAS::FieldCategories, 221
- FFLAS::MMHelperAlgo, 222
- FFLAS::ModeCategories, 222
- FFLAS::ParSeqHelper, 222
- FFLAS::Protected, 223
  - computeFactorClassic, 226
  - DotProdBoundClassic, 227
  - DynamicPeeling, 228
  - DynamicPeeling2, 229
  - fgemm\_convert, 230
  - fgemv\_convert, 232
  - fgemv\_convert, 233
  - fsquareCommon, 232
  - fsyrk\_convert, 233
  - igemm, 237
  - igemm\_colmajor, 236, 237
  - MatF2MatD\_Triangular, 237
  - MatF2MatFI\_Triangular, 238
  - min\_types, 235, 236
  - NeedDoublePreAddReduction, 231
  - NeedPreAddReduction, 230
  - NeedPreAxpPyReduction, 234
  - NeedPreScalReduction, 234
  - NeedPreSubReduction, 231

- ScalAndReduce, [232](#), [233](#)
- TRSMBound, [227](#)
- unfit, [236](#)
- WinogradCalc, [229](#)
- WinogradSteps, [228](#)
- WinogradThreshold, [227](#), [228](#)
- FFLAS::sell\_details, [238](#)
- FFLAS::sparse\_details, [238](#)
  - fspmm, [245–248](#)
  - fspmm\_dispatch, [244](#), [245](#)
  - fspmv, [242–244](#), [253](#), [254](#)
  - fspmv\_dispatch, [242](#)
  - init\_y, [241](#)
  - pfspmm, [249–251](#)
  - pfspmm\_dispatch, [248](#)
  - pfspmv, [252](#), [253](#)
- FFLAS::sparse\_details\_impl, [254](#)
  - fspmm, [262](#), [263](#), [270](#), [271](#), [277](#), [281](#), [282](#), [291](#)
  - fspmm\_mone, [264](#), [272](#), [282](#), [283](#)
  - fspmm\_mone\_simd\_aligned, [265](#), [272](#), [284](#)
  - fspmm\_mone\_simd\_unaligned, [265](#), [273](#), [284](#)
  - fspmm\_one, [264](#), [271](#), [282](#), [283](#)
  - fspmm\_one\_simd\_aligned, [264](#), [272](#), [283](#)
  - fspmm\_one\_simd\_unaligned, [264](#), [272](#), [283](#)
  - fspmm\_simd\_aligned, [263](#), [271](#)
  - fspmm\_simd\_unaligned, [263](#), [271](#)
  - fspmv, [265](#), [266](#), [273](#), [277](#), [278](#), [284](#), [285](#), [287](#), [288](#), [291–294](#)
  - fspmv\_mone, [266](#), [274](#), [285](#), [288](#), [289](#), [295](#)
  - fspmv\_mone\_simd, [289](#), [295](#)
  - fspmv\_one, [266](#), [274](#), [285](#), [288](#), [294](#), [295](#)
  - fspmv\_one\_simd, [289](#), [295](#)
  - fspmv\_simd, [287](#), [288](#), [294](#)
  - pfspmm, [267](#), [274–276](#), [278](#), [279](#), [289](#), [290](#)
  - pfspmm\_mone, [268](#)
  - pfspmm\_one, [267](#), [268](#)
  - pfspmm\_zo, [279](#), [280](#)
  - pfspmv, [268](#), [269](#), [276](#), [280](#), [286](#), [290](#), [292](#)
  - pfspmv\_mone, [269](#), [270](#), [281](#), [286](#), [287](#), [293](#)
  - pfspmv\_one, [269](#), [270](#), [281](#), [286](#), [287](#), [293](#)
  - pfspmv\_task, [269](#)
- FFLAS::StrategyParameter, [296](#)
- FFLAS::StructureHelper, [296](#)
- FFLAS::vectorised, [296](#)
  - add, [299](#)
  - addp, [298](#)
  - axpyp, [299](#), [300](#)
  - modp, [302](#)
  - reduce, [300](#), [301](#)
  - scalp, [302](#)
  - sub, [299](#)
  - subp, [299](#)
  - VEC\_ADD, [298](#)
  - VEC\_SUB, [298](#)
- FFLAS::vectorised::unswitch, [303](#)
  - axpyp, [303](#), [304](#)
  - modp, [304](#)
  - scalp, [304](#), [305](#)
- fflas\_101.C, [907](#)
  - main, [907](#)
- fflas\_101\_lvl1.C, [907](#)
  - main, [908](#)
- FFLAS\_BASE
  - FFLAS, [81](#)
- fflas\_bounds.inl, [908](#)
  - \_\_FFLASFFPACK\_fflas\_bounds\_INL, [908](#)
  - FFLAS\_INT\_TYPE, [908](#)
- fflas\_c.h, [909](#)
  - fadd\_1\_modular\_double, [915](#)
  - fadd\_2\_modular\_double, [919](#)
  - faddin\_1\_modular\_double, [915](#)
  - faddin\_2\_modular\_double, [919](#)
  - fassign\_1\_modular\_double, [914](#)
  - fassign\_2\_modular\_double, [916](#)
  - faxpy\_1\_modular\_double, [914](#)
  - faxpy\_2\_modular\_double, [918](#)
  - fdot\_1\_modular\_double, [915](#)
  - fequal\_1\_modular\_double, [914](#)
  - fequal\_2\_modular\_double, [916](#)
  - FFLAS\_C\_BASE, [912](#)
  - FFLAS\_C\_DIAG, [912](#)
  - FFLAS\_C\_ORDER, [911](#)
  - FFLAS\_C\_SIDE, [912](#)
  - FFLAS\_C\_TRANSPOSE, [911](#)
  - FFLAS\_C\_UPLO, [911](#)
  - FFLAS\_COMPILED, [911](#)
  - FflasColMajor, [911](#)
  - FflasDouble, [912](#)
  - FflasFloat, [912](#)
  - FflasGeneric, [912](#)
  - FflasLeft, [912](#)
  - FflasLower, [912](#)
  - FflasNonUnit, [912](#)
  - FflasNoTrans, [911](#)
  - FflasRight, [912](#)
  - FflasRowMajor, [911](#)
  - FflasTrans, [911](#)
  - FflasUnit, [912](#)
  - FflasUpper, [912](#)
  - fgemm\_3\_modular\_double, [921](#)
  - fgemv\_2\_modular\_double, [920](#)
  - fger\_2\_modular\_double, [920](#)
  - fidentity\_2\_modular\_double, [917](#)
  - fiszero\_1\_modular\_double, [913](#)
  - fiszero\_2\_modular\_double, [916](#)
  - fmove\_2\_modular\_double, [918](#)
  - fneg\_1\_modular\_double, [913](#)
  - fneg\_2\_modular\_double, [917](#)
  - fnegin\_1\_modular\_double, [913](#)
  - fnegin\_2\_modular\_double, [917](#)
  - freduce\_1\_modular\_double, [913](#)
  - freduce\_2\_modular\_double, [917](#)
  - freducein\_1\_modular\_double, [912](#)
  - freducein\_2\_modular\_double, [917](#)
  - fscal\_1\_modular\_double, [914](#)
  - fscal\_2\_modular\_double, [918](#)

- fscaln\_1\_modular\_double, 914
- fscaln\_2\_modular\_double, 918
- fsquare\_3\_modular\_double, 921
- fsub\_1\_modular\_double, 915
- fsub\_2\_modular\_double, 919
- fsubin\_1\_modular\_double, 916
- fsubin\_2\_modular\_double, 919
- fswap\_1\_modular\_double, 915
- ftmm\_3\_modular\_double, 921
- ftsm\_3\_modular\_double, 920
- ftsv\_2\_modular\_double, 920
- fzero\_1\_modular\_double, 913
- fzero\_2\_modular\_double, 916
- FFLAS\_C\_BASE
  - fflas\_c.h, 912
- FFLAS\_C\_DIAG
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FFLAS\_C\_ORDER
  - fflas\_c.h, 911
  - ffpack\_c.h, 1031
- FFLAS\_C\_SIDE
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FFLAS\_C\_TRANSPOSE
  - fflas\_c.h, 911
  - ffpack\_c.h, 1032
- FFLAS\_C\_UPLO
  - fflas\_c.h, 911
  - ffpack\_c.h, 1032
- FFLAS\_COMPILED
  - fflas\_c.h, 911
  - ffpack\_inst.C, 1060
  - ffpack\_inst.h, 1062
- fflas\_const\_cast
  - FFPACK, 385, 386, 400
- fflas\_delete
  - FFLAS, 164, 165, 202
- FFLAS\_DIAG
  - FFLAS, 80
- FFLAS\_ELT
  - fflas\_L1\_inst.C, 949, 950
  - fflas\_L1\_inst.h, 950, 951
  - fflas\_L2\_inst.C, 953
  - fflas\_L2\_inst.h, 954
  - fflas\_L3\_inst.C, 957
  - fflas\_L3\_inst.h, 958
  - ffpack\_inst.C, 1061
  - ffpack\_inst.h, 1062
- fflas\_enum.h, 922
- fflas\_fadd.h, 922
- fflas\_fadd.inl, 924
  - \_\_FFLASFFPACK\_fadd\_INL, 925
- fflas\_fassign.h, 925
- fflas\_fassign.inl, 925
  - \_\_FFLASFFPACK\_fassign\_INL, 926
- fflas\_faxpy.inl, 926
  - \_\_FFLASFFPACK\_faxpy\_INL, 927
- fflas\_fdot.inl, 927
  - \_\_FFLASFFPACK\_fdot\_INL, 928
- fflas\_fgemm.inl, 928
  - \_\_FFLASFFPACK\_fgemm\_INL, 930
- fflas\_fgenv.inl, 930
  - \_\_FFLASFFPACK\_fgenv\_INL, 931
- fflas\_fgenv\_mp.inl, 932
  - \_\_FFLASFFPACK\_fgenv\_mp\_INL, 932
- fflas\_fger.inl, 932
  - \_\_FFLASFFPACK\_fger\_INL, 933
- fflas\_fger\_mp.inl, 934
  - \_\_FFPACK\_fger\_mp\_INL, 934
- FFLAS\_FIELD
  - fflas\_L1\_inst.C, 949
  - fflas\_L1\_inst.h, 950
  - fflas\_L2\_inst.C, 953
  - fflas\_L2\_inst.h, 954
  - fflas\_L3\_inst.C, 957
  - fflas\_L3\_inst.h, 958
  - ffpack\_inst.C, 1061
  - ffpack\_inst.h, 1062
- FFLAS\_FORMAT
  - FFLAS, 82
- fflas\_freduce.h, 934
- fflas\_freduce.inl, 935
  - \_\_FFLASFFPACK\_fflas\_freduce\_INL, 937
  - FFLASFFPACK\_COPY\_REDUCE, 937
- fflas\_freduce\_mp.inl, 937
  - \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL, 937
- fflas\_freivalds.inl, 937
  - \_\_FFLASFFPACK\_freivalds\_INL, 938
- fflas\_fscal.h, 938
- fflas\_fscal.inl, 938
  - \_\_FFLASFFPACK\_fscal\_INL, 939
- fflas\_fscal\_mp.inl, 940
  - \_\_FFLASFFPACK\_fscal\_mp\_INL, 940
- fflas\_fsyr2k.inl, 940
  - \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL, 941
- fflas\_fsyrk.inl, 941
  - \_\_FFLASFFPACK\_fflas\_fsyrk\_INL, 943
- fflas\_fsyrk\_strassen.inl, 943
  - \_\_FFLASFFPACK\_fflas\_fsyrk\_strassen\_INL, 944
- fflas\_ftmm.inl, 944
  - \_\_FFLASFFPACK\_ftmm\_INL, 944
- fflas\_ftsm.inl, 944
  - \_\_FFLASFFPACK\_ftsm\_INL, 945
- fflas\_ftsm\_mp.inl, 945
  - \_\_FFPACK\_ftsm\_mp\_INL, 946
- fflas\_ftsv.inl, 946
  - \_\_FFLASFFPACK\_ftsv\_INL, 946
- fflas\_helpers.inl, 946
  - \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL, 947
- FFLAS\_INT\_TYPE
  - fflas\_bounds.inl, 908
- fflas\_intrinsic.h, 948
- fflas\_io.h, 948
- fflas\_L1\_inst.C, 948
  - \_\_FFLAS\_L1\_INST\_C, 949

- FFLAS\_ELT, [949](#), [950](#)
- FFLAS\_FIELD, [949](#)
- INST\_OR\_DECL, [949](#)
- fflas\_L1\_inst.h, [950](#)
  - FFLAS\_ELT, [950](#), [951](#)
  - FFLAS\_FIELD, [950](#)
  - INST\_OR\_DECL, [950](#)
- fflas\_L1\_inst\_implement.inl, [951](#)
- fflas\_L2\_inst.C, [952](#)
  - \_\_FFLAS\_L2\_INST\_C, [952](#)
  - FFLAS\_ELT, [953](#)
  - FFLAS\_FIELD, [953](#)
  - INST\_OR\_DECL, [953](#)
- fflas\_L2\_inst.h, [953](#)
  - FFLAS\_ELT, [954](#)
  - FFLAS\_FIELD, [954](#)
  - INST\_OR\_DECL, [954](#)
- fflas\_L2\_inst\_implement.inl, [955](#)
- fflas\_L3\_inst.C, [956](#)
  - \_\_FFLAS\_L3\_INST\_C, [957](#)
  - FFLAS\_ELT, [957](#)
  - FFLAS\_FIELD, [957](#)
  - INST\_OR\_DECL, [957](#)
- fflas\_L3\_inst.h, [957](#)
  - FFLAS\_ELT, [958](#)
  - FFLAS\_FIELD, [958](#)
  - INST\_OR\_DECL, [958](#)
- fflas\_L3\_inst\_implement.inl, [959](#)
  - \_\_FFLAS\_\_TRSM\_READONLY, [959](#)
- fflas\_level1.inl, [960](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL, [962](#)
- fflas\_level2.inl, [962](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL, [964](#)
- fflas\_level3.inl, [965](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL, [967](#)
  - \_\_FFLAS\_\_TRSM\_READONLY, [967](#)
- fflas\_lv1.C, [967](#)
  - fadd\_1\_modular\_double, [971](#)
  - faddin\_1\_modular\_double, [971](#)
  - fassign\_1\_modular\_double, [969](#)
  - faxpy\_1\_modular\_double, [970](#)
  - fdot\_1\_modular\_double, [970](#)
  - fequal\_1\_modular\_double, [969](#)
  - fiszero\_1\_modular\_double, [969](#)
  - fneg\_1\_modular\_double, [969](#)
  - fnegin\_1\_modular\_double, [968](#)
  - freduce\_1\_modular\_double, [968](#)
  - freducein\_1\_modular\_double, [968](#)
  - fscal\_1\_modular\_double, [970](#)
  - fscalin\_1\_modular\_double, [970](#)
  - fsub\_1\_modular\_double, [971](#)
  - fsubin\_1\_modular\_double, [971](#)
  - fswap\_1\_modular\_double, [970](#)
  - fzero\_1\_modular\_double, [969](#)
- fflas\_lv2.C, [972](#)
  - fadd\_2\_modular\_double, [975](#)
  - faddin\_2\_modular\_double, [976](#)
  - fassign\_2\_modular\_double, [973](#)
  - faxpy\_2\_modular\_double, [975](#)
  - fequal\_2\_modular\_double, [973](#)
  - fgemv\_2\_modular\_double, [976](#)
  - fger\_2\_modular\_double, [977](#)
  - fidentity\_2\_modular\_double, [974](#)
  - fiszero\_2\_modular\_double, [973](#)
  - fmove\_2\_modular\_double, [975](#)
  - fneg\_2\_modular\_double, [974](#)
  - fnegin\_2\_modular\_double, [974](#)
  - freduce\_2\_modular\_double, [974](#)
  - freducein\_2\_modular\_double, [974](#)
  - fscal\_2\_modular\_double, [975](#)
  - fscalin\_2\_modular\_double, [975](#)
  - fsub\_2\_modular\_double, [976](#)
  - fsubin\_2\_modular\_double, [976](#)
  - ftsv\_2\_modular\_double, [977](#)
  - fzero\_2\_modular\_double, [973](#)
- fflas\_lv3.C, [977](#)
  - fgemm\_3\_modular\_double, [979](#)
  - fsquare\_3\_modular\_double, [979](#)
  - ftmm\_3\_modular\_double, [978](#)
  - ftsm\_3\_modular\_double, [978](#)
- fflas\_memory.h, [979](#)
- fflas\_new
  - FFLAS, [165](#), [166](#), [201](#)
- FFLAS\_ORDER
  - FFLAS, [79](#)
- fflas\_pfgemm.inl, [980](#)
  - \_\_FFLASFFPACK\_DIMKPENALTY, [980](#)
  - \_\_FFLASFFPACK\_SEQPARTHRESHOLD, [980](#)
  - \_\_FFLASFFPACK\_fflas\_pfgemm\_INL, [980](#)
- fflas\_pftsm.inl, [981](#)
  - \_\_FFLASFFPACK\_fflas\_pftsm\_INL, [981](#)
  - PTRSM\_HYBRID\_THRESHOLD, [981](#)
- fflas\_plevel1.h, [981](#)
- fflas\_randommatrix.h, [982](#)
- FFLAS\_SIDE
  - FFLAS, [81](#)
- fflas\_simd.h, [984](#)
  - CONST, [985](#)
  - FLOAT\_MOD, [985](#)
  - INLINE, [985](#)
  - NORML\_MOD, [985](#)
  - PURE, [985](#)
  - Simd, [986](#)
  - SIMD\_INT, [985](#)
- fflas\_sparse.C, [986](#)
- fflas\_sparse.h, [986](#)
  - \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE, [990](#)
  - assume\_aligned, [990](#)
  - DENSE\_THRESHOLD, [991](#)
  - index\_t, [990](#)
  - ROUND\_DOWN, [990](#)
- fflas\_sparse.inl, [991](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL, [993](#)
- FFLAS\_TRANSPOSE
  - FFLAS, [80](#)
- fflas\_transpose.h, [993](#)

- FFLAS\_TRANSPOSE\_BLOCKSIZE, 994
- LD, 994
- ST, 994
- FFLAS\_TRANSPOSE\_BLOCKSIZE
  - fflas\_transpose.h, 994
- FFLAS\_UPLO
  - FFLAS, 80
- FflasAuto
  - FFLAS, 82
- FflasBinary
  - FFLAS, 82
- FflasColMajor
  - FFLAS, 80
  - fflas\_c.h, 911
  - ffpack\_c.h, 1032
- FflasDense
  - FFLAS, 82
- FflasDouble
  - FFLAS, 81
  - fflas\_c.h, 912
- FFLASFFPACK\_abort
  - debug.h, 894
- FFLASFFPACK\_check
  - debug.h, 894
- FFLASFFPACK\_checkers\_fflas\_inl\_H
  - checkers\_fflas.inl, 864
- FFLASFFPACK\_checkers\_ffpack\_inl\_H
  - checkers\_ffpack.inl, 865
- FFLASFFPACK\_COPY\_REDUCE
  - fflas\_freduce.inl, 937
- FFLASFFPACK\_PERM\_BKSIZE
  - ffpack\_permutation.inl, 1071
- FflasFloat
  - FFLAS, 81
  - fflas\_c.h, 912
- FflasGeneric
  - FFLAS, 81
  - fflas\_c.h, 912
- FflasLeft
  - FFLAS, 81
  - fflas\_c.h, 912
  - ffpack\_c.h, 1033
- FflasLeftTri
  - FFLAS, 80
- FflasLower
  - FFLAS, 80
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FflasMaple
  - FFLAS, 82
- FflasMath
  - FFLAS, 82
- FflasNonUnit
  - FFLAS, 80
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FflasNoTrans
  - FFLAS, 80
- fflas\_c.h, 911
- ffpack\_c.h, 1032
- FflasRight
  - FFLAS, 81
  - fflas\_c.h, 912
  - ffpack\_c.h, 1033
- FflasRightTri
  - FFLAS, 80
- FflasRowMajor
  - FFLAS, 80
  - fflas\_c.h, 911
  - ffpack\_c.h, 1032
- FflasSageMath
  - FFLAS, 82
- FflasSMS
  - FFLAS, 82
- FflasTrans
  - FFLAS, 80
  - fflas\_c.h, 911
  - ffpack\_c.h, 1032
- FflasUnit
  - FFLAS, 80
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FflasUpper
  - FFLAS, 80
  - fflas\_c.h, 912
  - ffpack\_c.h, 1032
- FFPACK, 46, 305
  - \_PLUQ, 384
  - applyP, 323, 324, 386
  - applyP\_block, 378
  - Bruhat2EchelonPermutation, 366
  - buildMatrix, 371
  - CharPoly, 342, 343, 372, 391, 392
  - Checker\_charpoly, 322
  - Checker\_Det, 322
  - Checker\_invert, 322
  - Checker\_PLUQ, 322
  - chooseField, 416
  - chooseField< Givaro::ZRing< double > >, 417
  - chooseField< Givaro::ZRing< float > >, 416
  - chooseField< Givaro::ZRing< int32\_t > >, 416
  - chooseField< Givaro::ZRing< int64\_t > >, 416
  - ColRankProfileSubmatrix, 355, 397
  - ColRankProfileSubmatrixIndices, 354, 396
  - ColumnEchelonForm, 335, 336, 390
  - ColumnRankProfile, 351, 352, 396
  - composePermutationsLLL, 381
  - composePermutationsLLM, 382
  - composePermutationsMLM, 382
  - CompressToBlockBiDiagonal, 364
  - ComputeRPermutation, 366, 369
  - cyclic\_shift\_col, 383, 386
  - cyclic\_shift\_mathPerm, 382
  - cyclic\_shift\_row, 383, 386
  - cyclic\_shift\_row\_col, 382, 386
  - Danilevski, 371

- Det, [347](#), [372](#), [394](#)
- doApplyS, [378](#)
- doApplyT, [380](#)
- ExpandBlockBiDiagonalToBruhat, [365](#)
- expandLCRE, [369](#)
- failure, [400](#)
- fflas\_const\_cast, [385](#), [386](#), [400](#)
- fgesv, [327](#), [328](#), [387](#), [388](#)
- fgetrs, [326](#), [387](#)
- ForceCheck\_charpoly, [323](#)
- ForceCheck\_Det, [322](#)
- ForceCheck\_invert, [323](#)
- ForceCheck\_PLUQ, [322](#)
- fsytrf, [331](#), [332](#)
- fsytrf\_BC\_Crout, [373](#)
- fsytrf\_BC\_RL, [373](#)
- fsytrf\_LOW\_RPM\_BC\_Crout, [373](#)
- fsytrf\_nonunit, [332](#), [374](#)
- fsytrf\_RPM, [375](#)
- fsytrf\_UP\_RPM, [374](#)
- fsytrf\_UP\_RPM\_BC\_Crout, [374](#)
- fsytrf\_UP\_RPM\_BC\_RL, [373](#)
- ftssyr2k, [330](#)
- ftstr, [330](#)
- fttri, [329](#), [388](#)
- fttrm, [329](#), [388](#)
- getEchelonForm, [357](#), [358](#)
- getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT >  
>, [398](#)
- getEchelonTransform, [359](#)
- getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT  
> >, [398](#)
- getLTBruhatGen, [363](#)
- getReducedEchelonForm, [360](#), [361](#)
- getReducedEchelonForm< FFLAS\_FIELD<  
FFLAS\_ELT > >, [399](#)
- getReducedEchelonTransform, [361](#)
- getReducedEchelonTransform< FFLAS\_FIELD<  
FFLAS\_ELT > >, [399](#)
- getTriangular, [356](#), [357](#)
- getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >,  
[397](#)
- getTridiagonal, [375](#)
- Invert, [340](#), [341](#), [391](#)
- Invert2, [341](#), [391](#)
- isOdd, [400](#), [401](#)
- IsSingular, [346](#), [394](#)
- KrylovElim, [393](#)
- LAPACKPerm2MathPerm, [323](#)
- LeadingSubmatrixRankProfiles, [353](#)
- LQUPtoInverseOfFullRankMinor, [367](#), [400](#)
- LTBruhatGen, [362](#)
- LTQSorter, [364](#)
- LUdivine, [334](#), [376](#), [389](#)
- LUdivine\_gauss, [375](#), [389](#)
- LUdivine\_small, [375](#), [389](#)
- MathPerm2LAPACKPerm, [323](#)
- MatrixApplyS, [378](#), [379](#)
- MatrixApplyT, [380](#), [381](#)
- MatVecMinPoly, [344](#), [393](#)
- MinPoly, [344](#), [392](#)
- MonotonicApplyP, [325](#)
- MonotonicCompress, [376](#)
- MonotonicCompressCycles, [377](#)
- MonotonicCompressMorePivots, [377](#)
- MonotonicExpand, [377](#)
- NonZeroRandomMatrix, [401](#)
- NullSpaceBasis, [349](#), [395](#)
- pColumnEchelonForm, [336](#)
- pColumnRankProfile, [352](#)
- pDet, [347](#)
- PermApplyS, [379](#)
- PermApplyT, [381](#)
- PLUQ, [333](#), [334](#), [384](#), [385](#), [389](#)
- PLUQ\_basecaseCrout, [384](#)
- PLUQ\_basecaseV2, [384](#)
- PLUQ\_basecaseV3, [383](#)
- PLUQtoEchelonPermutation, [362](#)
- pPLUQ, [333](#)
- pRank, [345](#)
- pReducedColumnEchelonForm, [338](#)
- pReducedRowEchelonForm, [339](#)
- productBruhatxTS, [367](#), [370](#)
- pRowEchelonForm, [337](#)
- pRowRankProfile, [351](#)
- pSolve, [349](#)
- RandInt, [404](#)
- RandomIndexSubset, [406](#)
- RandomLTQSMatixWithRankandQSorter, [416](#)
- RandomLTQSRankProfileMatrix, [408](#)
- RandomMatrix, [402](#)
- RandomMatrixWithDet, [414](#)
- RandomMatrixWithRank, [405](#), [406](#)
- RandomMatrixWithRankandRandomRPM, [411](#)
- RandomMatrixWithRankandRPM, [408](#), [409](#)
- RandomNullSpaceVector, [349](#), [368](#), [395](#)
- RandomPermutation, [407](#)
- RandomRankProfileMatrix, [407](#)
- RandomSymmetricMatrix, [404](#)
- RandomSymmetricMatrixWithRankandRandom-  
RPM, [412](#)
- RandomSymmetricMatrixWithRankandRPM, [409](#),  
[410](#)
- RandomSymmetricRankProfileMatrix, [407](#)
- RandomTriangularMatrix, [403](#), [404](#)
- Rank, [345](#), [346](#), [393](#)
- RankProfileFromLU, [352](#)
- ReducedColumnEchelonForm, [337](#), [338](#), [390](#)
- ReducedRowEchelonForm, [339](#), [340](#), [390](#)
- RowEchelonForm, [336](#), [337](#), [390](#)
- RowRankProfile, [350](#), [351](#), [396](#)
- RowRankProfileSubmatrix, [355](#), [397](#)
- RowRankProfileSubmatrixIndices, [353](#), [396](#)
- Solve, [348](#), [394](#)
- solveLB, [368](#), [395](#)
- solveLB2, [369](#), [395](#)

- SpecRankProfile, 393
- swapval, 407
- threads\_fgemm, 385
- threads\_ftrsm, 385
- TInverter, 366, 369
- trinv\_left, 329, 388
- ffpack-fgesv.C, 994
  - main, 994
- ffpack-solve.C, 994
  - main, 995
- ffpack.C, 995
  - applyP\_modular\_double, 1000
  - ColRankProfileSubmatrix\_modular\_double, 1014
  - ColRankProfileSubmatrixIndices\_modular\_double, 1013
  - ColumnEchelonForm\_modular\_double, 1003
  - ColumnEchelonForm\_modular\_float, 1004
  - ColumnEchelonForm\_modular\_int32\_t, 1005
  - ColumnRankProfile\_modular\_double, 1012
  - composePermutationsLLL, 1000
  - composePermutationsLLM, 999
  - composePermutationsMLM, 1000
  - cyclic\_shift\_col\_modular\_double, 1000
  - cyclic\_shift\_mathPerm, 1000
  - cyclic\_shift\_row\_modular\_double, 1000
  - Det\_modular\_double, 1011
  - fgesv\_modular\_double, 1002
  - fgesvin\_modular\_double, 1001
  - fgetrsin\_modular\_double, 1001
  - fgetrsv\_modular\_double, 1001
  - ftrtri\_modular\_double, 1002
  - ftrtrm\_modular\_double, 1002
  - getEchelonForm\_modular\_double, 1014
  - getEchelonFormin\_modular\_double, 1015
  - getEchelonTransform\_modular\_double, 1015
  - getReducedEchelonForm\_modular\_double, 1015
  - getReducedEchelonFormin\_modular\_double, 1016
  - getReducedEchelonTransform\_modular\_double, 1016
  - getTriangular\_modular\_double, 1014
  - getTriangularin\_modular\_double, 1014
  - Invert2\_modular\_double, 1010
  - Invert\_modular\_double, 1009
  - Invertin\_modular\_double, 1009
  - IsSingular\_modular\_double, 1010
  - KrylovElim\_modular\_double, 1010
  - LAPACKPerm2MathPerm, 998
  - LeadingSubmatrixRankProfiles, 1013
  - LUdivine\_modular\_double, 1003
  - MathPerm2LAPACKPerm, 998
  - MatrixApplyS\_modular\_double, 998
  - MatrixApplyT\_modular\_double, 999
  - NullSpaceBasis\_modular\_double, 1012
  - pColumnEchelonForm\_modular\_double, 1006
  - pColumnEchelonForm\_modular\_float, 1007
  - pColumnEchelonForm\_modular\_int32\_t, 1008
  - PermApplyS\_double, 999
  - PermApplyT\_double, 999
  - PLUQ\_modular\_double, 1002
  - PLUQtoEchelonPermutation, 1016
  - pReducedColumnEchelonForm\_modular\_double, 1007
  - pReducedColumnEchelonForm\_modular\_float, 1008
  - pReducedColumnEchelonForm\_modular\_int32\_t, 1009
  - pReducedRowEchelonForm\_modular\_double, 1007
  - pReducedRowEchelonForm\_modular\_float, 1008
  - pReducedRowEchelonForm\_modular\_int32\_t, 1009
  - pRowEchelonForm\_modular\_double, 1006
  - pRowEchelonForm\_modular\_float, 1007
  - pRowEchelonForm\_modular\_int32\_t, 1008
  - RandomNullSpaceVector\_modular\_double, 1012
  - Rank\_modular\_double, 1010
  - RankProfileFromLU, 1013
  - ReducedColumnEchelonForm\_modular\_double, 1004
  - ReducedColumnEchelonForm\_modular\_float, 1005
  - ReducedColumnEchelonForm\_modular\_int32\_t, 1006
  - ReducedRowEchelonForm\_modular\_double, 1004
  - ReducedRowEchelonForm\_modular\_float, 1005
  - ReducedRowEchelonForm\_modular\_int32\_t, 1006
  - RowEchelonForm\_modular\_double, 1003
  - RowEchelonForm\_modular\_float, 1004
  - RowEchelonForm\_modular\_int32\_t, 1005
  - RowRankProfile\_modular\_double, 1012
  - RowRankProfileSubmatrix\_modular\_double, 1013
  - RowRankProfileSubmatrixIndices\_modular\_double, 1013
  - Solve\_modular\_double, 1011
  - solveLB2\_modular\_double, 1011
  - solveLB\_modular\_double, 1011
  - SpecRankProfile\_modular\_double, 1010
  - trinv\_left\_modular\_double, 1002
- ffpack.doxy, 1016
- ffpack.h, 1016
  - \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD, 1025
  - \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD, 1025
- ffpack.inl, 1026
  - \_\_FFLASFFPACK\_ffpack\_INL, 1027
- FFPACK::Protected, 417
  - ArithProg, 421
  - CompressRows, 422
  - CompressRowsQA, 423
  - CompressRowsQK, 423
  - Danilevski, 421
  - DeCompressRows, 423
  - DeCompressRowsQA, 424
  - DeCompressRowsQK, 423
  - fgemv\_kgf, 420

- GaussJordan, 419
- Hybrid\_KGF\_LUK\_MinPoly, 422
- KellerGehrig, 420
- KGFast, 420
- KGFast\_generalized, 420
- LUdivine\_construct, 418, 424
- LUKrylov, 420
- LUKrylov\_KGFast, 421
- MatVecMinPoly, 422
- newD, 422
- RandomKrylovPrecond, 421
- updatedD, 422
- ffpack\_bruhatgen.inl, 1027
- \_\_FFLASFFPACK\_ffpack\_bruhatgen\_inl, 1028
- ffpack\_c.h, 1028
- applyP\_modular\_double, 1036
- ColRankProfileSubmatrix\_modular\_double, 1047
- ColRankProfileSubmatrixIndices\_modular\_double, 1046
- ColumnEchelonForm\_modular\_double, 1039
- ColumnEchelonForm\_modular\_float, 1039
- ColumnEchelonForm\_modular\_int32\_t, 1040
- ColumnRankProfile\_modular\_double, 1045
- composePermutationsLLL, 1035
- composePermutationsLLM, 1035
- composePermutationsMLM, 1035
- cyclic\_shift\_col\_modular\_double, 1035
- cyclic\_shift\_mathPerm, 1035
- cyclic\_shift\_row\_modular\_double, 1035
- Det\_modular\_double, 1044
- FFLAS\_C\_DIAG, 1032
- FFLAS\_C\_ORDER, 1031
- FFLAS\_C\_SIDE, 1032
- FFLAS\_C\_TRANSPOSE, 1032
- FFLAS\_C\_UPLO, 1032
- FflasColMajor, 1032
- FflasLeft, 1033
- FflasLower, 1032
- FflasNonUnit, 1032
- FflasNoTrans, 1032
- FflasRight, 1033
- FflasRowMajor, 1032
- FflasTrans, 1032
- FflasUnit, 1032
- FflasUpper, 1032
- FFPACK\_C\_CHARPOLY\_TAG, 1033
- FFPACK\_C\_LU\_TAG, 1033
- FFPACK\_C\_MINPOLY\_TAG, 1033
- FFPACK\_COMPILED, 1031
- FfpackArithProg, 1033
- FfpackDanilevski, 1033
- FfpackDense, 1033
- FfpackHybrid, 1033
- FfpackKG, 1033
- FfpackKGF, 1033
- FfpackKGFast, 1033
- FfpackKGFastG, 1033
- FfpackLUK, 1033
- FfpackSingular, 1033
- FfpackSlabRecursive, 1033
- FfpackTileRecursive, 1033
- fgesv\_modular\_double, 1037
- fgesvin\_modular\_double, 1037
- fgetrs\_modular\_double, 1036
- fgetrsin\_modular\_double, 1036
- ftrtri\_modular\_double, 1037
- ftrtrm\_modular\_double, 1037
- getEchelonForm\_modular\_double, 1047
- getEchelonFormin\_modular\_double, 1048
- getEchelonTransform\_modular\_double, 1048
- getReducedEchelonForm\_modular\_double, 1048
- getReducedEchelonFormin\_modular\_double, 1049
- getReducedEchelonTransform\_modular\_double, 1049
- getTriangular\_modular\_double, 1047
- getTriangularin\_modular\_double, 1047
- Invert2\_modular\_double, 1043
- Invert\_modular\_double, 1042
- Invertin\_modular\_double, 1042
- IsSingular\_modular\_double, 1044
- KrylovElim\_modular\_double, 1043
- LAPACKPerm2MathPerm, 1033
- LeadingSubmatrixRankProfiles, 1046
- LUdivine\_gauss\_modular\_double, 1038
- LUdivine\_modular\_double, 1038
- LUdivine\_small\_modular\_double, 1038
- MathPerm2LAPACKPerm, 1034
- MatrixApplyS\_modular\_double, 1034
- MatrixApplyT\_modular\_double, 1034
- NullSpaceBasis\_modular\_double, 1045
- PermApplyS\_double, 1034
- PermApplyT\_double, 1034
- PLUQ\_modular\_double, 1038
- PLUQtoEchelonPermutation, 1049
- RandomNullSpaceVector\_modular\_double, 1045
- Rank\_modular\_double, 1043
- RankProfileFromLU, 1046
- ReducedColumnEchelonForm\_modular\_double, 1040
- ReducedColumnEchelonForm\_modular\_float, 1041
- ReducedColumnEchelonForm\_modular\_int32\_t, 1041
- ReducedRowEchelonForm2\_modular\_double, 1042
- ReducedRowEchelonForm\_modular\_double, 1040
- ReducedRowEchelonForm\_modular\_float, 1041
- ReducedRowEchelonForm\_modular\_int32\_t, 1041
- REF\_modular\_double, 1042
- RowEchelonForm\_modular\_double, 1039
- RowEchelonForm\_modular\_float, 1039
- RowEchelonForm\_modular\_int32\_t, 1040
- RowRankProfile\_modular\_double, 1045
- RowRankProfileSubmatrix\_modular\_double, 1046

- RowRankProfileSubmatrixIndices\_modular\_double, 1046
- Solve\_modular\_double, 1044
- solveLB2\_modular\_double, 1044
- solveLB\_modular\_double, 1044
- SpecRankProfile\_modular\_double, 1043
- trinv\_left\_modular\_double, 1037
- FFPACK\_C\_CHARPOLY\_TAG
  - ffpack\_c.h, 1033
- FFPACK\_C\_LU\_TAG
  - ffpack\_c.h, 1033
- FFPACK\_C\_MINPOLY\_TAG
  - ffpack\_c.h, 1033
- ffpack\_charpoly.inl, 1049
  - \_\_FFLASFFPACK\_charpoly\_INL, 1050
- ffpack\_charpoly\_danilevski.inl, 1050
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL, 1050
- ffpack\_charpoly\_kgfast.inl, 1051
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL, 1051
- ffpack\_charpoly\_kgfastgeneralized.inl, 1051
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL, 1052
- ffpack\_charpoly\_kglu.inl, 1052
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL, 1052
- ffpack\_charpoly\_mp.inl, 1052
  - \_\_FFPACK\_charpoly\_mp\_INL, 1053
- FFPACK\_COMPILED
  - ffpack\_c.h, 1031
- ffpack\_det\_mp.inl, 1053
  - \_\_FFPACK\_det\_mp\_INL, 1053
- ffpack\_echelonforms.inl, 1054
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, 1055
  - \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL, 1055
- ffpack\_fgesv.inl, 1055
  - \_\_FFLASFFPACK\_ffpack\_fgesv\_INL, 1055
- ffpack\_fgetrs.inl, 1056
  - \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL, 1056
- ffpack\_frobenius.inl, 1056
- ffpack\_fsytrf.inl, 1057
  - \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL, 1058
- ffpack\_ftrssyr2k.inl, 1058
  - \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL, 1058
- ffpack\_ftrstr.inl, 1059
  - \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL, 1059
- ffpack\_ftrtr.inl, 1059
  - \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL, 1060
- ENABLE\_ALL\_CHECKINGS, 1060
- ffpack\_inst.C, 1060
  - \_\_FFPACK\_INST\_C, 1060
  - FFLAS\_COMPILED, 1060
  - FFLAS\_ELT, 1061
  - FFLAS\_FIELD, 1061
  - INST\_OR\_DECL, 1060
- ffpack\_inst.h, 1061
  - FFLAS\_COMPILED, 1062
  - FFLAS\_ELT, 1062
  - FFLAS\_FIELD, 1062
  - INST\_OR\_DECL, 1062
- ffpack\_inst\_implem.inl, 1063
- ffpack\_invert.inl, 1066
  - \_\_FFLASFFPACK\_ffpack\_invert\_INL, 1066
- ffpack\_krylovelim.inl, 1066
  - \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL, 1066
- ffpack\_ludivine.inl, 1067
  - \_\_FFLASFFPACK\_ffpack\_ludivine\_INL, 1067
- ffpack\_ludivine\_mp.inl, 1068
  - \_\_FFPACK\_ludivine\_mp\_INL, 1068
- ffpack\_minpoly.inl, 1068
  - \_\_FFLASFFPACK\_ffpack\_minpoly\_INL, 1069
- ffpack\_permutation.inl, 1069
  - \_\_FFLASFFPACK\_ffpack\_permutation\_INL, 1071
- FFLASFFPACK\_PERM\_BKSIZE, 1071
- ffpack\_pluq.inl, 1071
  - \_\_FFLASFFPACK\_ffpack\_pluq\_INL, 1072
- CROUT, 1072
- ffpack\_pluq.inl, 1072
  - \_\_FFPACK\_pluq\_mp\_INL, 1073
- ffpack\_ppluq.inl, 1073
  - \_\_FFLASFFPACK\_ffpack\_ppluq\_INL, 1073
  - \_\_FFLAS\_\_TRSM\_READONLY, 1073
- PBASECASE\_K, 1073
- ffpack\_rankprofiles.inl, 1074
  - \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL, 1075
- FpackArithProg
  - ffpack\_c.h, 1033
- FpackDanilevski
  - ffpack\_c.h, 1033
- FpackDense
  - ffpack\_c.h, 1033
- FpackHybrid
  - ffpack\_c.h, 1033
- FpackKG
  - ffpack\_c.h, 1033
- FpackKGF
  - ffpack\_c.h, 1033
- FpackKGFast
  - ffpack\_c.h, 1033
- FpackKGFastG
  - ffpack\_c.h, 1033
- FpackLUK
  - ffpack\_c.h, 1033
- FpackSingular
  - ffpack\_c.h, 1033
- FpackSlabRecursive
  - ffpack\_c.h, 1033
- FpackTileRecursive
  - ffpack\_c.h, 1033
- fgemm
  - FFLAS, 94–101, 153, 189, 190
- fgemm\_3\_modular\_double
  - fflas\_c.h, 921

- fflas\_lvl3.C, [979](#)
- fgemm\_classical.inl, [1075](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL, [1075](#)
- fgemm\_classical\_mp.inl, [1075](#)
  - \_\_FFPACK\_fgemm\_classical\_INL, [1077](#)
- fgemm\_convert
  - FFLAS::Protected, [230](#)
- fgemm\_winograd.inl, [1077](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL, [1078](#)
- NEWWINO, [1078](#)
- fgemv
  - FFLAS, [104–109](#), [185](#), [197](#)
- fgemv\_2\_modular\_double
  - fflas\_c.h, [920](#)
  - fflas\_lvl2.C, [976](#)
- fgemv\_convert
  - FFLAS::Protected, [232](#)
- fgemv\_kgf
  - FFPACK::Protected, [420](#)
- fger
  - FFLAS, [109–113](#), [186](#)
- fger\_2\_modular\_double
  - fflas\_c.h, [920](#)
  - fflas\_lvl2.C, [977](#)
- fger\_convert
  - FFLAS::Protected, [233](#)
- fgesv
  - FFPACK, [327](#), [328](#), [387](#), [388](#)
- fgesv\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1037](#)
- fgesvin\_modular\_double
  - ffpack.C, [1001](#)
  - ffpack\_c.h, [1037](#)
- fgetrs
  - FFPACK, [326](#), [387](#)
- fgetrs\_modular\_double
  - ffpack\_c.h, [1036](#)
- fgetrsin\_modular\_double
  - ffpack.C, [1001](#)
  - ffpack\_c.h, [1036](#)
- fgetrsv\_modular\_double
  - ffpack.C, [1001](#)
- fidentity
  - FFLAS, [145](#), [146](#), [178](#)
- fidentity\_2\_modular\_double
  - fflas\_c.h, [917](#)
  - fflas\_lvl2.C, [974](#)
- Field
  - Bench< Elt >, [433](#)
  - benchmark-fgemm-rns.C, [838](#)
  - benchmark-pluq.C, [852](#)
  - FieldSimd< \_Field >, [475](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [784](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [787](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [791](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [799](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
  - Test< Elt >, [811](#)
  - test-compressQ.C, [1129](#)
- field
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [431](#)
  - associatedDelayedField< const Givaro::Modular< T, X > >, [431](#)
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, [432](#)
  - associatedDelayedField< const Givaro::ZRing< T > >, [432](#)
  - associatedDelayedField< Field >, [430](#)
- field-traits.h, [1079](#)
- field.doxy, [1081](#)
- FieldMax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- FieldMin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [533](#)
- FieldSimd
  - FieldSimd< \_Field >, [476](#)
- FieldSimd< \_Field >, [474](#)
  - add, [477](#)
  - add\_r, [477](#)
  - addin, [477](#)
  - addin\_r, [477](#)
  - alignment, [480](#)
  - axpy, [479](#)
  - axpy\_r, [479](#), [480](#)
  - axpyin, [479](#)
  - axpyin\_r, [480](#)
  - Element, [475](#)
  - Field, [475](#)
  - FieldSimd, [476](#)
  - init, [476](#)
  - maxpy, [480](#)
  - maxpyin, [480](#)
  - mod, [478](#)
  - mul, [478](#)
  - mul\_r, [479](#)
  - mulin, [479](#)
  - operator=, [476](#)
  - scalar\_t, [476](#)
  - simd, [475](#)
  - sub, [477](#)
  - sub\_r, [478](#)
  - subin, [478](#)
  - subin\_r, [478](#)
  - vect\_size, [480](#)

- vect\_t, [476](#)
- zero, [478](#)
- FieldTraits< FFPACK::RNSInteger< T > >, [481](#)
  - balanced, [482](#)
  - category, [481](#)
- FieldTraits< FFPACK::RNSIntegerMod< T > >, [482](#)
  - balanced, [482](#)
  - category, [482](#)
- FieldTraits< Field >, [481](#)
  - balanced, [481](#)
  - category, [481](#)
- FieldTraits< Givaro::Modular< Element > >, [482](#)
  - balanced, [483](#)
  - category, [483](#)
- FieldTraits< Givaro::ModularBalanced< Element > >, [483](#)
  - balanced, [483](#)
  - category, [483](#)
- FieldTraits< Givaro::ZRing< double > >, [483](#)
  - balanced, [484](#)
  - category, [484](#)
- FieldTraits< Givaro::ZRing< float > >, [484](#)
  - balanced, [484](#)
  - category, [484](#)
- FieldTraits< Givaro::ZRing< Givaro::Integer > >, [484](#)
  - balanced, [485](#)
  - category, [485](#)
- FieldTraits< Givaro::ZRing< int16\_t > >, [485](#)
  - balanced, [485](#)
  - category, [485](#)
- FieldTraits< Givaro::ZRing< int32\_t > >, [486](#)
  - balanced, [486](#)
  - category, [486](#)
- FieldTraits< Givaro::ZRing< int64\_t > >, [486](#)
  - balanced, [486](#)
  - category, [486](#)
- FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >, [487](#)
  - balanced, [487](#)
  - category, [487](#)
- FieldTraits< Givaro::ZRing< uint16\_t > >, [487](#)
  - balanced, [488](#)
  - category, [487](#)
- FieldTraits< Givaro::ZRing< uint32\_t > >, [488](#)
  - balanced, [488](#)
  - category, [488](#)
- FieldTraits< Givaro::ZRing< uint64\_t > >, [488](#)
  - balanced, [489](#)
  - category, [488](#)
- fill\_value
  - benchmark-fgemv.C, [842](#)
- findArgument
  - args-parser.h, [826](#)
- finit
  - FFLAS, [115](#), [117](#), [138](#), [146](#), [168](#), [179](#)
- finit\_rns
  - FFLAS, [165](#), [167](#)
- finit\_trans\_rns
  - FFLAS, [165](#)
- first\_component
  - Compose< H1, H2 >, [450](#)
- firstBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [492](#)
- fiszero
  - FFLAS, [141](#), [145](#), [170](#), [177](#)
- fiszero\_1\_modular\_double
  - fflas\_c.h, [913](#)
  - fflas\_lvl1.C, [969](#)
- fiszero\_2\_modular\_double
  - fflas\_c.h, [916](#)
  - fflas\_lvl2.C, [973](#)
- Fixed, [489](#)
- FixedPrecIntTag, [489](#)
- flimits.h, [1081](#)
  - in\_range, [1082](#)
- FLOAT\_MOD
  - fflas\_simd.h, [985](#)
- FloatingPointTestDistribution
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >::FloatingPointTestDistribution, [489](#)
- Floats
  - benchmark-dgemm.C, [831](#)
- floor
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [595](#)
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
- fma
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [596](#)
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [597](#)
- fmadd
  - ScalFunctions< Element >, [592](#)
  - Simd128\_impl< true, true, false, 2 >, [608](#)
  - Simd128\_impl< true, true, false, 4 >, [618](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [637](#)
  - Simd128\_impl< true, true, true, 4 >, [647](#)
  - Simd128\_impl< true, true, true, 8 >, [657](#)
  - Simd256\_impl< true, false, true, 8 >, [669](#)
  - Simd256\_impl< true, true, false, 2 >, [680](#)
  - Simd256\_impl< true, true, false, 4 >, [697](#), [698](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [718](#)
  - Simd256\_impl< true, true, true, 4 >, [730](#), [737](#)
  - Simd256\_impl< true, true, true, 8 >, [747](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
  - Simd512\_impl< true, true, false, 8 >, [768](#)
  - Simd512\_impl< true, true, true, 8 >, [777](#)
- fmaddin
  - ScalFunctions< Element >, [592](#)
  - Simd128\_impl< true, true, false, 2 >, [608](#)

[illegible]

- Simd256\_impl< true, true, false, 2 >, [677](#)
- Simd256\_impl< true, true, false, 4 >, [689](#), [691](#)
- Simd256\_impl< true, true, false, 8 >, [706](#)
- Simd256\_impl< true, true, true, 2 >, [719](#)
- Simd256\_impl< true, true, true, 4 >, [731](#), [738](#)
- Simd256\_impl< true, true, true, 8 >, [748](#)
- Simd512\_impl< true, true, false, 8 >, [765](#)
- Simd512\_impl< true, true, true, 8 >, [779](#)
- fneg
  - FFLAS, [139](#), [147](#), [169](#), [180](#)
- fneg\_1\_modular\_double
  - fflas\_c.h, [913](#)
  - fflas\_lvl1.C, [969](#)
- fneg\_2\_modular\_double
  - fflas\_c.h, [917](#)
  - fflas\_lvl2.C, [974](#)
- fnegin
  - FFLAS, [139](#), [147](#), [169](#), [179](#)
- fnegin\_1\_modular\_double
  - fflas\_c.h, [913](#)
  - fflas\_lvl1.C, [968](#)
- fnegin\_2\_modular\_double
  - fflas\_c.h, [917](#)
  - fflas\_lvl2.C, [974](#)
- fnmadd
  - ScalFunctions< Element >, [592](#)
  - Simd128\_impl< true, true, false, 2 >, [608](#)
  - Simd128\_impl< true, true, false, 4 >, [618](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [638](#)
  - Simd128\_impl< true, true, true, 4 >, [647](#)
  - Simd128\_impl< true, true, true, 8 >, [658](#)
  - Simd256\_impl< true, false, true, 8 >, [669](#)
  - Simd256\_impl< true, true, false, 2 >, [680](#)
  - Simd256\_impl< true, true, false, 4 >, [698](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [718](#)
  - Simd256\_impl< true, true, true, 4 >, [730](#), [737](#)
  - Simd256\_impl< true, true, true, 8 >, [748](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
  - Simd512\_impl< true, true, false, 8 >, [768](#)
  - Simd512\_impl< true, true, true, 8 >, [778](#)
- fnmaddin
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [608](#)
  - Simd128\_impl< true, true, false, 4 >, [618](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [638](#)
  - Simd128\_impl< true, true, true, 4 >, [648](#)
  - Simd128\_impl< true, true, true, 8 >, [658](#)
  - Simd256\_impl< true, false, true, 8 >, [669](#)
  - Simd256\_impl< true, true, false, 2 >, [680](#)
  - Simd256\_impl< true, true, false, 4 >, [698](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [719](#)
  - Simd256\_impl< true, true, true, 4 >, [730](#), [737](#)
  - Simd256\_impl< true, true, true, 8 >, [748](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
- Simd512\_impl< true, true, false, 8 >, [768](#)
- Simd512\_impl< true, true, true, 8 >, [778](#)
- fnmaddx
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [598](#)
  - Simd128\_impl< true, true, false, 2 >, [605](#)
  - Simd128\_impl< true, true, false, 4 >, [615](#)
  - Simd128\_impl< true, true, false, 8 >, [625](#)
  - Simd128\_impl< true, true, true, 2 >, [638](#)
  - Simd128\_impl< true, true, true, 4 >, [648](#)
  - Simd128\_impl< true, true, true, 8 >, [658](#)
  - Simd256\_impl< true, true, false, 2 >, [676](#)
  - Simd256\_impl< true, true, false, 4 >, [688](#), [691](#)
  - Simd256\_impl< true, true, false, 8 >, [706](#)
  - Simd256\_impl< true, true, true, 2 >, [719](#)
  - Simd256\_impl< true, true, true, 4 >, [730](#), [738](#)
  - Simd256\_impl< true, true, true, 8 >, [748](#)
  - Simd512\_impl< true, true, false, 8 >, [764](#)
  - Simd512\_impl< true, true, true, 8 >, [778](#)
- fnmaddxin
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [598](#)
  - Simd128\_impl< true, true, false, 2 >, [605](#)
  - Simd128\_impl< true, true, false, 4 >, [615](#)
  - Simd128\_impl< true, true, false, 8 >, [625](#)
  - Simd128\_impl< true, true, true, 2 >, [638](#)
  - Simd128\_impl< true, true, true, 4 >, [648](#)
  - Simd128\_impl< true, true, true, 8 >, [658](#)
  - Simd256\_impl< true, true, false, 2 >, [677](#)
  - Simd256\_impl< true, true, false, 4 >, [688](#), [691](#)
  - Simd256\_impl< true, true, false, 8 >, [706](#)
  - Simd256\_impl< true, true, true, 2 >, [719](#)
  - Simd256\_impl< true, true, true, 4 >, [731](#), [738](#)
  - Simd256\_impl< true, true, true, 8 >, [748](#)
  - Simd512\_impl< true, true, false, 8 >, [765](#)
  - Simd512\_impl< true, true, true, 8 >, [778](#)
- FOR1D
  - parallel.h, [1097](#)
- FOR2D
  - parallel.h, [1097](#)
- FORBLOCK1D
  - parallel.h, [1096](#)
- FORBLOCK2D
  - parallel.h, [1097](#)
- ForceCheck\_charpoly
  - FFPACK, [323](#)
- ForceCheck\_Det
  - FFPACK, [322](#)
- ForceCheck\_fgemm
  - FFLAS, [76](#)
- ForceCheck\_ftsm
  - FFLAS, [76](#)
- ForceCheck\_invert
  - FFPACK, [323](#)
- ForceCheck\_PLUQ
  - FFPACK, [322](#)
- ForStrategy1D
  - ForStrategy1D< blocksize\_t, Cut, Param >, [490](#)

- ForStrategy1D< blocksize\_t, Cut, Param >, [490](#)
  - begin, [491](#)
  - blockindex, [491](#)
  - build, [491](#)
  - changeBS, [492](#)
  - current, [491](#)
  - end, [491](#)
  - firstBlockSize, [492](#)
  - ForStrategy1D, [490](#)
  - ibeg, [491](#)
  - iend, [491](#)
  - initialize, [491](#)
  - isTerminated, [491](#)
  - lastBlockSize, [492](#)
  - numBlock, [492](#)
  - numblocks, [491](#)
  - operator++, [491](#)
- ForStrategy2D
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- ForStrategy2D< blocksize\_t, Cut, Param >, [492](#)
  - \_ibeg, [494](#)
  - \_iend, [494](#)
  - \_jbeg, [494](#)
  - \_jend, [494](#)
  - blockindex, [494](#)
  - BLOCKS, [495](#)
  - changeCBS, [495](#)
  - changeRBS, [495](#)
  - colblockindex, [494](#)
  - colBlockSize, [495](#)
  - colnumblocks, [494](#)
  - current, [495](#)
  - ForStrategy2D, [493](#)
  - ibegin, [493](#)
  - iend, [493](#)
  - initialize, [493](#)
  - isTerminated, [493](#)
  - jbegin, [493](#)
  - jend, [493](#)
  - lastCBS, [495](#)
  - lastRBS, [495](#)
  - numColBlock, [495](#)
  - numRowBlock, [495](#)
  - operator<<, [494](#)
  - operator++, [493](#)
  - rowblockindex, [494](#)
  - rowBlockSize, [494](#)
  - rownumblocks, [493](#)
- frand
  - FFLAS, [140](#), [144](#)
- freduce
  - FFLAS, [113–116](#), [118](#), [167](#), [168](#), [178](#), [179](#)
  - FFLAS::details, [215](#), [216](#)
- freduce\_1\_modular\_double
  - fflas\_c.h, [913](#)
  - fflas\_lvl1.C, [968](#)
- freduce\_2\_modular\_double
  - fflas\_c.h, [917](#)
  - fflas\_lvl2.C, [974](#)
- freduce\_constoverride
  - FFLAS, [114](#), [117](#)
- freducein\_1\_modular\_double
  - fflas\_c.h, [912](#)
  - fflas\_lvl1.C, [968](#)
- freducein\_2\_modular\_double
  - fflas\_c.h, [917](#)
  - fflas\_lvl2.C, [974](#)
- freivalds
  - FFLAS, [118](#)
- fscal
  - FFLAS, [119–123](#), [172](#), [181](#)
  - FFLAS::details, [216](#), [217](#)
- fscal\_1\_modular\_double
  - fflas\_c.h, [914](#)
  - fflas\_lvl1.C, [970](#)
- fscal\_2\_modular\_double
  - fflas\_c.h, [918](#)
  - fflas\_lvl2.C, [975](#)
- fscalin
  - FFLAS, [119–123](#), [172](#), [180](#)
  - FFLAS::details, [216](#), [217](#)
- fscalin\_1\_modular\_double
  - fflas\_c.h, [914](#)
  - fflas\_lvl1.C, [970](#)
- fscalin\_2\_modular\_double
  - fflas\_c.h, [918](#)
  - fflas\_lvl2.C, [975](#)
- fspmm
  - FFLAS, [164](#)
  - FFLAS::sparse\_details, [245–248](#)
  - FFLAS::sparse\_details\_impl, [262](#), [263](#), [270](#), [271](#), [277](#), [281](#), [282](#), [291](#)
- fspmm\_dispatch
  - FFLAS::sparse\_details, [244](#), [245](#)
- fspmm\_mone
  - FFLAS::sparse\_details\_impl, [264](#), [272](#), [282](#), [283](#)
- fspmm\_mone\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [265](#), [272](#), [284](#)
- fspmm\_mone\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [265](#), [273](#), [284](#)
- fspmm\_one
  - FFLAS::sparse\_details\_impl, [264](#), [271](#), [282](#), [283](#)
- fspmm\_one\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [264](#), [272](#), [283](#)
- fspmm\_one\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [264](#), [272](#), [283](#)
- fspmm\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [263](#), [271](#)
- fspmm\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [263](#), [271](#)
- fspmv
  - FFLAS, [162](#), [163](#)
  - FFLAS::sparse\_details, [242–244](#), [253](#), [254](#)
  - FFLAS::sparse\_details\_impl, [265](#), [266](#), [273](#), [277](#), [278](#), [284](#), [285](#), [287](#), [288](#), [291–294](#)
- fspmv\_dispatch

- FFLAS::sparse\_details, 242
- fspmv\_mone
  - FFLAS::sparse\_details\_impl, 266, 274, 285, 288, 289, 295
- fspmv\_mone\_simd
  - FFLAS::sparse\_details\_impl, 289, 295
- fspmv\_one
  - FFLAS::sparse\_details\_impl, 266, 274, 285, 288, 294, 295
- fspmv\_one\_simd
  - FFLAS::sparse\_details\_impl, 289, 295
- fspmv\_simd
  - FFLAS::sparse\_details\_impl, 287, 288, 294
- fsquare
  - FFLAS, 102, 103, 191
- fsquare\_3\_modular\_double
  - fflas\_c.h, 921
  - fflas\_lvl3.C, 979
- fsquareCommon
  - FFLAS::Protected, 232
- fsub
  - FFLAS, 84, 86, 175, 183
- fsub\_1\_modular\_double
  - fflas\_c.h, 915
  - fflas\_lvl1.C, 971
- fsub\_2\_modular\_double
  - fflas\_c.h, 919
  - fflas\_lvl2.C, 976
- fsubin
  - FFLAS, 84, 87, 184
- fsubin\_1\_modular\_double
  - fflas\_c.h, 916
  - fflas\_lvl1.C, 971
- fsubin\_2\_modular\_double
  - fflas\_c.h, 919
  - fflas\_lvl2.C, 976
- fswap
  - FFLAS, 142, 174
- fswap\_1\_modular\_double
  - fflas\_c.h, 915
  - fflas\_lvl1.C, 970
- fsyr2k
  - FFLAS, 124
- fsyrk
  - FFLAS, 125–130, 132, 133
- fsyrk.C, 1082
  - CUBE, 1082
  - GFOPS, 1083
  - main, 1083
  - TTimer, 1083
- fsyrk\_convert
  - FFLAS::Protected, 233
- fsyrk\_strassen
  - FFLAS, 133, 151
- fsytrf
  - FFPACK, 331, 332
- fsytrf.C, 1083
  - CUBE, 1083
- GFOPS, 1084
  - main, 1084
- TTimer, 1084
- fsytrf\_BC\_Crout
  - FFPACK, 373
- fsytrf\_BC\_RL
  - FFPACK, 373
- fsytrf\_LOW\_RPM\_BC\_Crout
  - FFPACK, 373
- fsytrf\_nonunit
  - FFPACK, 332, 374
- fsytrf\_RPM
  - FFPACK, 375
- fsytrf\_UP\_RPM
  - FFPACK, 374
- fsytrf\_UP\_RPM\_BC\_Crout
  - FFPACK, 374
- fsytrf\_UP\_RPM\_BC\_RL
  - FFPACK, 373
- ftrmm
  - FFLAS, 133, 134, 188
- ftrmm\_3\_modular\_double
  - fflas\_c.h, 921
  - fflas\_lvl3.C, 978
- ftrmmLeftLowerNoTransNonUnit< Element >, 495
- ftrmmLeftLowerNoTransUnit< Element >, 496
- ftrmmLeftLowerTransNonUnit< Element >, 496
- ftrmmLeftLowerTransUnit< Element >, 496
- ftrmmLeftUpperNoTransNonUnit< Element >, 496
- ftrmmLeftUpperNoTransUnit< Element >, 496
- ftrmmLeftUpperTransNonUnit< Element >, 496
- ftrmmLeftUpperTransUnit< Element >, 496
- ftrmmRightLowerNoTransNonUnit< Element >, 496
- ftrmmRightLowerNoTransUnit< Element >, 497
- ftrmmRightLowerTransNonUnit< Element >, 497
- ftrmmRightLowerTransUnit< Element >, 497
- ftrmmRightUpperNoTransNonUnit< Element >, 497
- ftrmmRightUpperNoTransUnit< Element >, 497
- ftrmmRightUpperTransNonUnit< Element >, 497
- ftrmmRightUpperTransUnit< Element >, 497
- ftrmv
  - FFLAS, 150
- ftrsm
  - FFLAS, 135, 136, 150, 154, 187
- ftrsm\_3\_modular\_double
  - fflas\_c.h, 920
  - fflas\_lvl3.C, 978
- ftrsmLeftLowerNoTransNonUnit< Element >, 497
- ftrsmLeftLowerNoTransUnit< Element >, 498
- ftrsmLeftLowerTransNonUnit< Element >, 498
- ftrsmLeftLowerTransUnit< Element >, 498
- ftrsmLeftUpperNoTransNonUnit< Element >, 498
- ftrsmLeftUpperNoTransUnit< Element >, 498
- ftrsmLeftUpperTransNonUnit< Element >, 499
- ftrsmLeftUpperTransUnit< Element >, 499
- ftrsmRightLowerNoTransNonUnit< Element >, 499
- ftrsmRightLowerNoTransUnit< Element >, 499
- ftrsmRightLowerTransNonUnit< Element >, 499

- ftsmRightLowerTransUnit< Element >, [499](#)
- ftsmRightUpperNoTransNonUnit< Element >, [499](#)
- ftsmRightUpperNoTransUnit< Element >, [499](#)
- ftsmRightUpperTransNonUnit< Element >, [500](#)
- ftsmRightUpperTransUnit< Element >, [500](#)
- ftssyr2k
  - FFPACK, [330](#)
- ftstr
  - FFPACK, [330](#)
- ftsv
  - FFLAS, [137](#), [186](#)
- ftsv\_2\_modular\_double
  - fflas\_c.h, [920](#)
  - fflas\_lvl2.C, [977](#)
- fttri
  - FFPACK, [329](#), [388](#)
- fttri.C, [1084](#)
  - CUBE, [1084](#)
  - GFOPS, [1085](#)
  - main, [1085](#)
  - TTimer, [1085](#)
- fttri\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1037](#)
- fttrm
  - FFPACK, [329](#), [388](#)
- fttrm\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1037](#)
- fzero
  - FFLAS, [140](#), [143](#), [170](#), [176](#)
- fzero\_1\_modular\_double
  - fflas\_c.h, [913](#)
  - fflas\_lvl1.C, [969](#)
- fzero\_2\_modular\_double
  - fflas\_c.h, [916](#)
  - fflas\_lvl2.C, [973](#)
- gather
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [686](#), [689](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [714](#)
  - Simd256\_impl< true, true, true, 4 >, [726](#), [733](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, false, true, 8 >, [754](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- GaussJordan
  - FFPACK::Protected, [419](#)
- GCC\_VERSION
  - fflas-ffpack-config.h, [904](#)
- gebp
  - FFLAS::details, [220](#)
- genData
  - benchmark-fgemv.C, [842](#)
- GenericTag, [500](#)
- genInputs
  - ScalFunctions< Element >, [590](#)
- genInputsWithZero
  - ScalFunctions< Element >, [590](#)
- get
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
- get\_default\_random\_generator
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [595](#)
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [597](#)
- getArgumentValue
  - FFLAS, [198](#)
- getDataType
  - FFLAS, [160](#), [161](#)
- getEchelonForm
  - FFPACK, [357](#), [358](#)
- getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [398](#)
- getEchelonForm\_modular\_double
  - ffpack.C, [1014](#)
  - ffpack\_c.h, [1047](#)
- getEchelonFormin\_modular\_double
  - ffpack.C, [1015](#)
  - ffpack\_c.h, [1048](#)
- getEchelonTransform
  - FFPACK, [359](#)
- getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [398](#)
- getEchelonTransform\_modular\_double
  - ffpack.C, [1015](#)
  - ffpack\_c.h, [1048](#)
- getListArgs
  - args-parser.h, [826](#)
- getLTBruhatGen
  - FFPACK, [363](#)
- getReducedEchelonForm
  - FFPACK, [360](#), [361](#)
- getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [399](#)
- getReducedEchelonForm\_modular\_double
  - ffpack.C, [1015](#)
  - ffpack\_c.h, [1048](#)
- getReducedEchelonFormin\_modular\_double
  - ffpack.C, [1016](#)
  - ffpack\_c.h, [1049](#)
- getReducedEchelonTransform

- FFPACK, [361](#)
- getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [399](#)
- getReducedEchelonTransform\_modular\_double
  - ffpack.C, [1016](#)
  - ffpack\_c.h, [1049](#)
- getSeed
  - FFLAS, [203](#)
- getStat
  - FFLAS, [163](#)
- getStatus
  - TestOneMethod< Simd >, [815](#)
- getTestName
  - TestOneMethod< Simd >, [815](#)
- getTLBSize
  - FFLAS, [202](#)
- getTriangular
  - FFPACK, [356](#), [357](#)
- getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [397](#)
- getTriangular\_modular\_double
  - ffpack.C, [1014](#)
  - ffpack\_c.h, [1047](#)
- getTriangularin\_modular\_double
  - ffpack.C, [1014](#)
  - ffpack\_c.h, [1047](#)
- getTridiagonal
  - FFPACK, [375](#)
- gf2ModularBalanced
  - regression-check.C, [1106](#)
- GFOPS
  - arithprog.C, [827](#)
  - autotune/charpoly.C, [859](#)
  - autotune/pluq.C, [1103](#)
  - fsyrk.C, [1083](#)
  - fsytrf.C, [1084](#)
  - fttrtri.C, [1085](#)
  - winograd.C, [1183](#)
- Givaro, [424](#)
- GRAIN
  - benchmark-fgemm-rns.C, [839](#)
- Grain, [500](#)
- greater
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [604](#)
  - Simd128\_impl< true, true, false, 4 >, [614](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [705](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [731](#), [738](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- greater\_eq
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [604](#)
  - Simd128\_impl< true, true, false, 4 >, [614](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 2 >, [676](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [705](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [732](#), [739](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
  - Simd512\_impl< true, true, false, 8 >, [764](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- hadd
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd512\_impl< true, false, true, 8 >, [759](#)
- hadd\_to\_scal
  - Simd128\_impl< true, true, false, 2 >, [605](#)
  - Simd128\_impl< true, true, false, 4 >, [615](#)
  - Simd128\_impl< true, true, false, 8 >, [625](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd256\_impl< true, true, false, 2 >, [677](#)
  - Simd256\_impl< true, true, false, 4 >, [689](#), [692](#)
  - Simd256\_impl< true, true, false, 8 >, [706](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [732](#), [739](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [759](#)
  - Simd512\_impl< true, true, false, 8 >, [765](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- half\_t
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [686](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [724](#), [725](#)
  - Simd256\_impl< true, true, true, 8 >, [742](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [772](#)
- has\_equal
  - FFLAS, [77](#)
- has\_minus
  - FFLAS, [77](#)
- has\_minus\_eq
  - FFLAS, [78](#)
- has\_minus\_eq\_impl< C >, [500](#)
  - value, [500](#)

- has\_minus\_impl< C >, [501](#)
  - value, [501](#)
- has\_mul
  - FFLAS, [78](#)
- has\_mul\_eq
  - FFLAS, [78](#)
- has\_mul\_eq\_impl< C >, [501](#)
  - value, [501](#)
- has\_mul\_impl< C >, [501](#)
  - value, [501](#)
- has\_operation< T >, [502](#)
  - value, [502](#)
- has\_plus
  - FFLAS, [77](#)
- has\_plus\_eq
  - FFLAS, [77](#)
- has\_plus\_eq\_impl< C >, [502](#)
  - value, [502](#)
- has\_plus\_impl< C >, [502](#)
  - value, [503](#)
- HAVE\_BLAS
  - config.h, [874](#)
- HAVE\_CBLAS
  - config.h, [874](#)
- HAVE\_CXX11
  - config.h, [874](#)
- HAVE\_DLFCN\_H
  - config.h, [874](#)
- HAVE\_FLOAT\_H
  - config.h, [874](#)
- HAVE\_INT128
  - config.h, [874](#)
- HAVE\_INTPTR\_T
  - config.h, [874](#)
- HAVE\_LAPACK
  - config.h, [875](#)
- HAVE\_LIMITS\_H
  - config.h, [875](#)
- HAVE\_LITTLE\_ENDIAN
  - config.h, [875](#)
- HAVE\_MEMORY\_H
  - config.h, [875](#)
- HAVE\_PTHREAD\_H
  - config.h, [875](#)
- HAVE\_STDDEF\_H
  - config.h, [875](#)
- HAVE\_STDINT\_H
  - config.h, [875](#)
- HAVE\_STDLIB\_H
  - config.h, [875](#)
- HAVE\_STRING\_H
  - config.h, [875](#)
- HAVE\_STRINGS\_H
  - config.h, [875](#)
- HAVE\_SYS\_STAT\_H
  - config.h, [875](#)
- HAVE\_SYS\_TIME\_H
  - config.h, [875](#)
- HAVE\_SYS\_TYPES\_H
  - config.h, [876](#)
- HAVE\_UNISTD\_H
  - config.h, [876](#)
- HelperFlag, [503](#)
  - aut, [503](#)
  - coo, [503](#)
  - csr, [503](#)
  - ell, [503](#)
  - none, [503](#)
  - pm1, [503](#)
- HelperMod
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [505](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [506](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [506](#)
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
  - HelperMod, [504](#)
  - invp, [504](#)
  - max, [504](#)
  - min, [504](#)
  - p, [504](#)
  - pow50rem, [504](#)
  - HelperMod< Field, ElementTraits >, [504](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [505](#)
  - HelperMod, [505](#)
  - p, [505](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [505](#)
  - HelperMod, [506](#)
  - p, [506](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [506](#)
  - HelperMod, [506](#)
  - invp, [507](#)
  - max, [507](#)
  - min, [507](#)
  - p, [506](#)
- helpString
  - Argument, [430](#)
- HYB\_ZO
  - FFLAS, [82](#)
- hyb\_zo.h, [1085](#)
- hyb\_zo\_pspmm.inl, [1085](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL, [1086](#)
- hyb\_zo\_pspmv.inl, [1086](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL, [1086](#)
- hyb\_zo\_spm.inl, [1086](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm\_INL, [1087](#)

- hyb\_zo\_spmv.inl, [1087](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL, [1087](#)
- hyb\_zo\_utils.inl, [1088](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL, [1088](#)
- Hybrid, [507](#)
- Hybrid\_KGF\_LUK\_MinPoly
  - FFPACK::Protected, [422](#)
- ibeg
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
- ibegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- idamax\_
  - config-blas.h, [869](#)
- iend
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- igebb11
  - FFLAS::details, [219](#)
- igebb14
  - FFLAS::details, [218](#)
- igebb21
  - FFLAS::details, [218](#)
- igebb24
  - FFLAS::details, [217](#)
- igebb41
  - FFLAS::details, [218](#)
- igebb44
  - FFLAS::details, [217](#)
- igebp
  - FFLAS::details, [219](#)
- igemm
  - FFLAS::Protected, [237](#)
- igemm.doxy, [1088](#)
- igemm.h, [1088](#)
- igemm.inl, [1089](#)
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL, [1089](#)
- igemm\_
  - FFLAS, [138](#)
- igemm\_colmajor
  - FFLAS::Protected, [236](#), [237](#)
- igemm\_kernels.h, [1089](#)
- igemm\_kernels.inl, [1090](#)
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL, [1091](#)
- igemm\_tools.h, [1091](#)
- igemm\_tools.inl, [1091](#)
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL, [1091](#)
- in\_range
  - flimits.h, [1082](#)
- index\_t
  - fflas\_sparse.h, [990](#)
  - parallel.h, [1095](#)
- InfNorm
  - FFLAS, [82](#)
- Info, [507](#), [508](#)
  - begin, [508](#), [510](#)
  - Info, [507](#)–[509](#)
  - operator=, [508](#), [509](#)
  - perm, [508](#), [510](#)
  - size, [508](#), [509](#)
- info
  - BlockTransposeSIMD< Field, Simd, >, [436](#)
- init
  - FieldSimd< \_Field >, [476](#)
  - rns\_double, [562](#), [563](#)
  - rns\_double\_extended, [575](#), [576](#)
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [584](#), [585](#)
- init\_transpose
  - rns\_double, [563](#)
- init\_y
  - FFLAS::sparse\_details, [241](#)
- initA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [532](#)
- initB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [532](#)
- initC
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [532](#)
- initialize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- initOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [532](#)
- INLINE
  - fflas\_simd.h, [985](#)
- inplace
  - Bench< Elt >, [435](#)
- inputs
  - TestOneMethod< Simd >, [815](#)
- INST\_OR\_DECL
  - fflas\_L1\_inst.C, [949](#)
  - fflas\_L1\_inst.h, [950](#)
  - fflas\_L2\_inst.C, [953](#)
  - fflas\_L2\_inst.h, [954](#)
  - fflas\_L3\_inst.C, [957](#)
  - fflas\_L3\_inst.h, [958](#)
  - ffpack\_inst.C, [1060](#)
  - ffpack\_inst.h, [1062](#)
- integer
  - rns\_double, [561](#)
  - rns\_double\_extended, [574](#)
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [583](#)
- Interfaces, [47](#)
- interfaces.doxy, [1092](#)
- IntType
  - ScalFunctionsBase< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >::FloatingPointTestDistribution, [489](#)

- inv
  - RNSIntegerMod< RNS >, [586](#)
- Invert
  - FFPACK, [340](#), [341](#), [391](#)
- Invert2
  - FFPACK, [341](#), [391](#)
- Invert2\_modular\_double
  - ffpack.C, [1010](#)
  - ffpack\_c.h, [1043](#)
- Invert\_modular\_double
  - ffpack.C, [1009](#)
  - ffpack\_c.h, [1042](#)
- Invertin\_modular\_double
  - ffpack.C, [1009](#)
  - ffpack\_c.h, [1042](#)
- invp
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineIntTag >, [507](#)
- is\_all\_same< Args >, [510](#)
- is\_all\_same< T, Args... >, [510](#)
  - value, [510](#)
- is\_all\_same<>, [510](#)
  - value, [510](#)
- is\_same\_element
  - Bench< Elt >, [433](#)
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [612](#)
  - Simd128\_impl< true, true, false, 8 >, [622](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [685](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [725](#)
  - Simd256\_impl< true, true, true, 8 >, [743](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [773](#)
  - Test< Elt >, [811](#)
- is\_simd< T >, [511](#)
  - type, [511](#)
  - value, [511](#)
- isMOne
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [584](#)
- isOdd
  - FFPACK, [400](#), [401](#)
- isOne
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [584](#)
- IsSingular
  - FFPACK, [346](#), [394](#)
- IsSingular\_modular\_double
  - ffpack.C, [1010](#)
  - ffpack\_c.h, [1044](#)
- isSparseMatrix< Field, M >, [511](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO > >, [511](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [512](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >, [512](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >, [512](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [513](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >, [513](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >, [513](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [513](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [514](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >, [514](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >, [514](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [515](#)
- isSparseMatrixMKLFormat< F, M >, [515](#)
- isSparseMatrixSimdFormat< F, M >, [515](#)
- isTerminated
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- isZero
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [584](#)
- isZOSparseMatrix< F, M >, [515](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [516](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [516](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [516](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [517](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [517](#)
- Iterative, [517](#)
- iters
  - Bench< Elt >, [435](#)
- jbegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
- kaapi\_routines.inl, [1092](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL, [1092](#)
- KellerGehrig

- FFPACK::Protected, [420](#)
- KGFast
  - FFPACK::Protected, [420](#)
- KGFast\_generalized
  - FFPACK::Protected, [420](#)
- kmax
  - Sparse< \_Field, SparseMatrix\_t::COO >, [785](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [791](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [795](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [799](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
- KrylovElim
  - FFPACK, [393](#)
- KrylovElim\_modular\_double
  - ffpack.C, [1010](#)
  - ffpack\_c.h, [1043](#)
- lapack.C, [1092](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [1092](#)
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, [1092](#)
  - main, [1092](#)
- LAPACKPerm2MathPerm
  - FFPACK, [323](#)
  - ffpack.C, [998](#)
  - ffpack\_c.h, [1033](#)
- lastBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [492](#)
- lastCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- lastRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [495](#)
- launch\_fger
  - test-fger.C, [1143](#)
- launch\_fger\_dispatch
  - test-fger.C, [1143](#)
- launch\_MM
  - test-fgemm.C, [1139](#)
- launch\_MM\_dispatch
  - test-fgemm-check.C, [1137](#)
  - test-fgemm.C, [1139](#)
- launch\_MV
  - test-fgemv.C, [1141](#)
- launch\_MV\_dispatch
  - test-fgemv.C, [1141](#)
- launch\_test
  - test-charpoly.C, [1128](#)
  - test-lu.C, [1166](#)
  - test-quasisep.C, [1174](#)
- launch\_wino
  - benchmark-wino.C, [855](#)
- LazyTag, [517](#)
- LD
  - fflas\_transpose.h, [994](#)
- ld
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [794](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
- LeadingSubmatrixRankProfiles
  - FFPACK, [353](#)
  - ffpack.C, [1013](#)
  - ffpack\_c.h, [1046](#)
- lesser
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [604](#)
  - Simd128\_impl< true, true, false, 4 >, [614](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 2 >, [676](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [705](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [732](#), [739](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- lesser\_eq
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [604](#)
  - Simd128\_impl< true, true, false, 4 >, [614](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [659](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 2 >, [676](#)
  - Simd256\_impl< true, true, false, 4 >, [688](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [705](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [732](#), [739](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, false, true, 8 >, [758](#)
  - Simd512\_impl< true, true, false, 8 >, [764](#)
  - Simd512\_impl< true, true, true, 8 >, [779](#)
- limits< char >, [518](#)
  - digits, [518](#)
  - max, [518](#)
  - min, [518](#)
  - T, [518](#)
- limits< double >, [518](#)
  - digits, [519](#)
  - max, [519](#)
  - min, [519](#)

- T, [519](#)
- limits< float >, [519](#)
  - digits, [520](#)
  - max, [520](#)
  - min, [520](#)
  - T, [519](#)
- limits< Givaro::Integer >, [520](#)
  - max, [520](#)
  - min, [520](#)
  - T, [520](#)
- limits< int >, [521](#)
  - digits, [521](#)
  - max, [521](#)
  - min, [521](#)
  - T, [521](#)
- limits< long >, [521](#)
  - digits, [522](#)
  - max, [522](#)
  - min, [522](#)
  - T, [522](#)
- limits< long long >, [522](#)
  - digits, [523](#)
  - max, [522](#)
  - min, [522](#)
  - T, [522](#)
- limits< Reclnt::rint< K > >, [523](#)
  - max, [523](#)
  - min, [523](#)
  - T, [523](#)
- limits< Reclnt::ruint< K > >, [523](#)
  - max, [524](#)
  - min, [524](#)
  - T, [524](#)
- limits< short int >, [524](#)
  - digits, [525](#)
  - max, [524](#)
  - min, [524](#)
  - T, [524](#)
- limits< signed char >, [525](#)
  - digits, [525](#)
  - max, [525](#)
  - min, [525](#)
  - T, [525](#)
- limits< T >, [518](#)
- limits< unsigned char >, [525](#)
  - digits, [526](#)
  - max, [526](#)
  - min, [526](#)
  - T, [526](#)
- limits< unsigned int >, [526](#)
  - digits, [527](#)
  - max, [527](#)
  - min, [527](#)
  - T, [526](#)
- limits< unsigned long >, [527](#)
  - digits, [527](#)
  - max, [527](#)
  - min, [527](#)
- T, [527](#)
- limits< unsigned long long >, [528](#)
  - digits, [528](#)
  - max, [528](#)
  - min, [528](#)
  - T, [528](#)
- limits< unsigned short int >, [528](#)
  - digits, [529](#)
  - max, [529](#)
  - min, [529](#)
  - T, [529](#)
- load
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [689](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [714](#)
  - Simd256\_impl< true, true, true, 4 >, [726](#), [733](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, false, true, 8 >, [754](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- loadu
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [689](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [714](#)
  - Simd256\_impl< true, true, true, 4 >, [726](#), [733](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, false, true, 8 >, [754](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- LQUPtoInverseOfFullRankMinor
  - FFPACK, [367](#), [400](#)
- LT\_OBJDIR
  - config.h, [876](#)
- LTBruhatGen
  - FFPACK, [362](#)
- LTQSorter
  - FFPACK, [364](#)
- LUdivine
  - FFPACK, [334](#), [376](#), [389](#)
- LUdivine\_construct
  - FFPACK::Protected, [418](#), [424](#)
- LUdivine\_gauss

- FFPACK, [375](#), [389](#)
- LUdivine\_gauss\_modular\_double
  - ffpack\_c.h, [1038](#)
- LUdivine\_modular\_double
  - ffpack.C, [1003](#)
  - ffpack\_c.h, [1038](#)
- LUdivine\_small
  - FFPACK, [375](#), [389](#)
- LUdivine\_small\_modular\_double
  - ffpack\_c.h, [1038](#)
- LUKrylov
  - FFPACK::Protected, [420](#)
- LUKrylov\_KGFast
  - FFPACK::Protected, [421](#)
- m
  - Bench< Elt >, [435](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [785](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [786](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [790](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [791](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [794](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [796](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [799](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [803](#)
- MachineFloatTag, [529](#)
- MachineIntTag, [529](#)
- main
  - 101-fgemm.C, [823](#)
  - 2x2-fgemm.C, [823](#)
  - 2x2-ftrsv.C, [824](#)
  - 2x2-pluq.C, [824](#)
  - arithprog.C, [828](#)
  - autotune/charpoly.C, [860](#)
  - autotune/pluq.C, [1103](#)
  - benchmark-charpoly-mp.C, [828](#)
  - benchmark-charpoly.C, [829](#)
  - benchmark-checkers.C, [830](#)
  - benchmark-dgemm.C, [831](#)
  - benchmark-dgetrf.C, [832](#)
  - benchmark-dgetri.C, [833](#)
  - benchmark-dsytrf.C, [834](#)
  - benchmark-dtrsm.C, [834](#)
  - benchmark-dtrtri.C, [835](#)
  - benchmark-fadd-lvl2.C, [836](#)
  - benchmark-fdot.C, [837](#)
  - benchmark-fgemm-mp.C, [838](#)
  - benchmark-fgemm-rns.C, [839](#)
  - benchmark-fgemm.C, [840](#)
  - benchmark-fgemv.C, [844](#)
  - benchmark-fgesv.C, [845](#)
  - benchmark-fsyr2k.C, [845](#)
  - benchmark-fsyrk.C, [846](#)
  - benchmark-fsytrf.C, [847](#)
  - benchmark-ftrsm-mp.C, [848](#)
  - benchmark-ftrsm.C, [848](#)
  - benchmark-ftrsv.C, [849](#)
  - benchmark-ftrtri.C, [850](#)
  - benchmark-inverse.C, [850](#)
  - benchmark-lqup-mp.C, [851](#)
  - benchmark-lqup.C, [851](#)
  - benchmark-pluq.C, [853](#)
  - benchmark-quasisep.C, [854](#)
  - benchmark-storage-transpose.C, [854](#)
  - benchmark-wino.C, [855](#)
  - cblas.C, [859](#)
  - clapack.C, [866](#)
  - cuda.C, [893](#)
  - det.C, [894](#)
  - examples/charpoly.C, [860](#)
  - examples/pluq.C, [1103](#)
  - fblas.C, [903](#)
  - fflas-101\_1.C, [903](#)
  - fflas-101\_3.C, [903](#)
  - fflas\_101.C, [907](#)
  - fflas\_101\_lvl1.C, [908](#)
  - ffpack-fgesv.C, [994](#)
  - ffpack-solve.C, [995](#)
  - fsyrk.C, [1083](#)
  - fsytrf.C, [1084](#)
  - ftrtri.C, [1085](#)
  - lapack.C, [1092](#)
  - matmul.C, [1093](#)
  - rank.C, [1104](#)
  - regression-check.C, [1106](#)
  - solve.C, [1125](#)
  - test-charpoly-check.C, [1128](#)
  - test-charpoly.C, [1129](#)
  - test-compressQ.C, [1130](#)
  - test-det-check.C, [1130](#)
  - test-det.C, [1131](#)
  - test-echelon.C, [1134](#)
  - test-fadd.C, [1135](#)
  - test-fdot.C, [1136](#)
  - test-fgemm-check.C, [1137](#)
  - test-fgemm.C, [1140](#)
  - test-fgemv.C, [1141](#)
  - test-fger.C, [1143](#)
  - test-fgesv.C, [1145](#)
  - test-finit.C, [1146](#)
  - test-fscal.C, [1147](#)
  - test-fsyr2k.C, [1148](#)
  - test-fsyrk.C, [1150](#)
  - test-fsytrf.C, [1152](#)
  - test-ftrmm.C, [1153](#)
  - test-ftrmv.C, [1154](#)
  - test-ftrsm-check.C, [1155](#)
  - test-ftrsm.C, [1156](#)
  - test-ftrssyr2k.C, [1158](#)
  - test-ftrstr.C, [1159](#)
  - test-ftrsv.C, [1160](#)

- test-ftrtri.C, 1161
- test-interfaces-c.c, 1162
- test-invert-check.C, 1162
- test-io.C, 1163
- test-lu.C, 1167
- test-maxdelayeddim.C, 1168
- test-minpoly.C, 1169
- test-multifile2.C, 1170
- test-nullspace.C, 1171
- test-permutations.C, 1172
- test-pluq-check.C, 1173
- test-quasisep.C, 1174
- test-rankprofiles.C, 1175
- test-rpm.C, 1176
- test-simd.C, 1179
- test-solve.C, 1180
- test-storage-transpose.C, 1181
- winograd.C, 1183
- mainpage.doxy, 1093
- mask\_high
  - Simd128\_impl< true, true, false, 8 >, 629
  - Simd128\_impl< true, true, true, 8 >, 659
  - Simd256\_impl< true, true, false, 8 >, 710
  - Simd256\_impl< true, true, true, 8 >, 749
  - Simd512\_impl< true, true, false, 8 >, 769
  - Simd512\_impl< true, true, true, 8 >, 779
- mask\_t
  - read\_sparse.h, 1105
- maskstore
  - Simd512\_impl< true, true, false, 8 >, 763
  - Simd512\_impl< true, true, true, 8 >, 774
- MatF2MatD\_Triangular
  - FFLAS::Protected, 237
- MatF2MatFI\_Triangular
  - FFLAS::Protected, 238
- MathPerm2LAPACKPerm
  - FFPACK, 323
  - ffpack.C, 998
  - ffpack\_c.h, 1034
- Matio.h, 1093
  - read\_field, 1093
  - write\_field, 1093
- matmul.C, 1093
  - main, 1093
- matmul.doxy, 1094
- Matrix Multiplication Algorithms, 45
- MatrixApplyS
  - FFPACK, 378, 379
- MatrixApplyS\_modular\_double
  - ffpack.C, 998
  - ffpack\_c.h, 1034
- MatrixApplyT
  - FFPACK, 380, 381
- MatrixApplyT\_modular\_double
  - ffpack.C, 999
  - ffpack\_c.h, 1034
- MatVecMinPoly
  - FFPACK, 344, 393
- FFPACK::Protected, 422
- max
  - HelperMod< Field, ElementCategories::MachineIntTag >, 504
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 507
  - limits< char >, 518
  - limits< double >, 519
  - limits< float >, 520
  - limits< Givaro::Integer >, 520
  - limits< int >, 521
  - limits< long >, 522
  - limits< long long >, 522
  - limits< Reclnt::rint< K > >, 523
  - limits< Reclnt::ruint< K > >, 524
  - limits< short int >, 524
  - limits< signed char >, 525
  - limits< unsigned char >, 526
  - limits< unsigned int >, 527
  - limits< unsigned long >, 527
  - limits< unsigned long long >, 528
  - limits< unsigned short int >, 529
- max3
  - FFLAS, 83
- max4
  - FFLAS, 83
- MAX\_THREADS
  - parallel.h, 1096
- MAX\_WITH\_SIZE\_T
  - test-maxdelayeddim.C, 1168
- maxCardinality
  - FFLAS, 164
- maxCardinality< Givaro::Modular< int32\_t > >
  - FFLAS, 164
- maxCardinality< Givaro::Modular< int64\_t > >
  - FFLAS, 164
- maxCol
  - StatsMatrix, 806
- maxColDifference
  - StatsMatrix, 807
- MaxDelayedDim
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 532
- maxElement
  - RNSIntegerMod< RNS >, 584
- maxpy
  - FieldSimd< \_Field >, 480
- maxpyin
  - FieldSimd< \_Field >, 480
- maxRow
  - StatsMatrix, 806
- maxrow
  - Sparse< \_Field, SparseMatrix\_t::COO >, 785
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 787
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 788
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 792

- Sparse< \_Field, SparseMatrix\_t::ELL >, [793](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [795](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [797](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [798](#)
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [800](#)
- Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- maxRowDifference
  - StatsMatrix, [807](#)
- MaxStorableValue
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [534](#)
- min
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [507](#)
  - limits< char >, [518](#)
  - limits< double >, [519](#)
  - limits< float >, [520](#)
  - limits< Givaro::Integer >, [520](#)
  - limits< int >, [521](#)
  - limits< long >, [522](#)
  - limits< long long >, [522](#)
  - limits< RecInt::rint< K > >, [523](#)
  - limits< RecInt::ruint< K > >, [524](#)
  - limits< short int >, [524](#)
  - limits< signed char >, [525](#)
  - limits< unsigned char >, [526](#)
  - limits< unsigned int >, [527](#)
  - limits< unsigned long >, [527](#)
  - limits< unsigned long long >, [528](#)
  - limits< unsigned short int >, [529](#)
- min3
  - FFLAS, [82](#)
- min4
  - FFLAS, [83](#)
- min\_types
  - FFLAS::Protected, [235](#), [236](#)
- minCardinality
  - FFLAS, [164](#)
- minCol
  - StatsMatrix, [807](#)
- minColDifference
  - StatsMatrix, [807](#)
- minElement
  - RNSIntegerMod< RNS >, [584](#)
- MinPoly
  - FFPACK, [344](#), [392](#)
- minRow
  - StatsMatrix, [806](#)
- minRowDifference
  - StatsMatrix, [807](#)
- MKL\_CONFIG, [424](#)
- MKLSparseMatrixFormat
  - FFLAS, [77](#)
- MMHelper
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [535](#), [536](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [538](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [540](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [541](#), [542](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [543](#), [544](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [531](#)
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [535](#)
  - MMHelper, [535](#), [536](#)
  - normA, [536](#)
  - normB, [536](#)
  - operator<<, [536](#)
  - parseq, [537](#)
  - recLevel, [537](#)
  - Self\_t, [535](#)
  - setNorm, [536](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [537](#)
  - MMHelper, [538](#)
  - normA, [539](#)
  - normB, [539](#)
  - operator<<, [538](#)
  - parseq, [539](#)
  - recLevel, [539](#)
  - Self\_t, [537](#)
  - setNorm, [538](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [539](#)
  - MMHelper, [540](#)
  - operator<<, [540](#)
  - parseq, [540](#)
  - recLevel, [540](#)
  - Self\_t, [539](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [541](#)
  - MMHelper, [541](#), [542](#)
  - normA, [542](#)
  - normB, [542](#)
  - operator<<, [542](#)
  - parseq, [543](#)
  - recLevel, [542](#)
  - Self\_t, [541](#)
  - setNorm, [542](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [543](#)
  - MMHelper, [543](#), [544](#)

- operator<<, [544](#)
- parseq, [544](#)
- recLevel, [544](#)
- Self\_t, [543](#)
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [530](#)
- Amax, [534](#)
- Amin, [534](#)
- Aunfit, [532](#)
- Bmax, [534](#)
- Bmin, [534](#)
- Bunfit, [532](#)
- checkA, [532](#)
- checkB, [533](#)
- checkOut, [533](#)
- Cmax, [534](#)
- Cmin, [534](#)
- DelayedField, [531](#)
- delayedField, [534](#)
- DelayedField\_t, [531](#)
- DFElt, [531](#)
- FieldMax, [534](#)
- FieldMin, [533](#)
- initA, [532](#)
- initB, [532](#)
- initC, [532](#)
- initOut, [532](#)
- MaxDelayedDim, [532](#)
- MaxStorableValue, [534](#)
- MMHelper, [531](#)
- operator<<, [533](#)
- Outmax, [534](#)
- Outmin, [534](#)
- parseq, [534](#)
- recLevel, [533](#)
- Self\_t, [531](#)
- setOutBounds, [532](#)
- mod
  - FieldSimd< \_Field >, [478](#)
  - Simd128\_impl< true, true, false, 2 >, [609](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [629](#)
  - Simd128\_impl< true, true, true, 2 >, [639](#)
  - Simd128\_impl< true, true, true, 4 >, [649](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd256\_impl< true, true, false, 2 >, [681](#)
  - Simd256\_impl< true, true, false, 4 >, [700](#)
  - Simd256\_impl< true, true, false, 8 >, [710](#)
  - Simd256\_impl< true, true, true, 2 >, [720](#)
  - Simd256\_impl< true, true, true, 4 >, [732](#), [739](#)
  - Simd256\_impl< true, true, true, 8 >, [750](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
- MODE
  - parallel.h, [1098](#)
- ModeTraits< Field >, [544](#)
  - value, [545](#)
- ModeTraits< Givaro::Modular< Element, Compute > >, [545](#)
  - value, [545](#)
- ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, [545](#)
  - value, [545](#)
- ModeTraits< Givaro::Modular< int16\_t, Compute > >, [546](#)
  - value, [546](#)
- ModeTraits< Givaro::Modular< int32\_t, Compute > >, [546](#)
  - value, [546](#)
- ModeTraits< Givaro::Modular< int64\_t, uint64\_t > >, [546](#)
  - value, [547](#)
- ModeTraits< Givaro::Modular< int8\_t, Compute > >, [547](#)
  - value, [547](#)
- ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >, [547](#)
  - value, [547](#)
- ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [547](#)
  - value, [548](#)
- ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [548](#)
  - value, [548](#)
- ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [548](#)
  - value, [548](#)
- ModeTraits< Givaro::ModularBalanced< Element > >, [549](#)
  - value, [549](#)
- ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [549](#)
  - value, [549](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t > >, [549](#)
  - value, [549](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t > >, [550](#)
  - value, [550](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t > >, [550](#)
  - value, [550](#)
- ModeTraits< Givaro::Montgomery< T > >, [550](#)
  - value, [551](#)
- ModeTraits< Givaro::ZRing< double > >, [551](#)
  - value, [551](#)
- ModeTraits< Givaro::ZRing< float > >, [551](#)
  - value, [551](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [551](#)
  - value, [552](#)
- ModField
  - rns\_double, [561](#)
  - rns\_double\_extended, [574](#)
  - RNSIntegerMod< RNS >, [583](#)
- modp

- FFLAS::vectorised, [302](#)
  - FFLAS::vectorised::unswitch, [304](#)
  - ModularBalanced< T >, [552](#)
  - ModularTag, [552](#)
  - mOne
    - RNSInteger< RNS >, [581](#)
    - RNSIntegerMod< RNS >, [588](#)
  - mone
    - FFLAS, [81](#)
    - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [800](#)
  - MonotonicApplyP
    - FFPACK, [325](#)
  - MonotonicCompress
    - FFPACK, [376](#)
  - MonotonicCompressCycles
    - FFPACK, [377](#)
  - MonotonicCompressMorePivots
    - FFPACK, [377](#)
  - MonotonicExpand
    - FFPACK, [377](#)
  - Montgomery< T >, [552](#)
  - mul
    - FieldSimd< \_Field >, [478](#)
    - RNSIntegerMod< RNS >, [586](#)
    - ScalFunctions< Element >, [592](#)
    - Simd128\_impl< true, true, false, 2 >, [608](#)
    - Simd128\_impl< true, true, false, 4 >, [618](#)
    - Simd128\_impl< true, true, false, 8 >, [628](#)
    - Simd128\_impl< true, true, true, 2 >, [637](#)
    - Simd128\_impl< true, true, true, 4 >, [647](#)
    - Simd128\_impl< true, true, true, 8 >, [657](#)
    - Simd256\_impl< true, false, true, 8 >, [668](#)
    - Simd256\_impl< true, true, false, 2 >, [680](#)
    - Simd256\_impl< true, true, false, 4 >, [697](#)
    - Simd256\_impl< true, true, false, 8 >, [709](#)
    - Simd256\_impl< true, true, true, 2 >, [718](#)
    - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
    - Simd256\_impl< true, true, true, 8 >, [747](#)
    - Simd512\_impl< true, false, true, 8 >, [757](#)
    - Simd512\_impl< true, true, false, 8 >, [768](#)
    - Simd512\_impl< true, true, true, 8 >, [777](#)
  - mul\_r
    - FieldSimd< \_Field >, [479](#)
  - mulhi
    - ScalFunctionsBase< Element, typename enable\_if<
      - is\_integral< Element >::value >::type >, [597](#)
 >
      - Simd128\_impl< true, true, false, 2 >, [604](#)
      - Simd128\_impl< true, true, false, 4 >, [614](#)
      - Simd128\_impl< true, true, false, 8 >, [624](#)
      - Simd128\_impl< true, true, true, 2 >, [637](#)
      - Simd128\_impl< true, true, true, 4 >, [647](#)
      - Simd128\_impl< true, true, true, 8 >, [657](#)
      - Simd256\_impl< true, true, false, 2 >, [676](#)
      - Simd256\_impl< true, true, false, 4 >, [688](#), [690](#)
      - Simd256\_impl< true, true, false, 8 >, [705](#)
      - Simd256\_impl< true, true, true, 2 >, [718](#)
      - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
      - Simd256\_impl< true, true, true, 8 >, [747](#)
  - Simd512\_impl< true, true, false, 8 >, [764](#)
  - Simd512\_impl< true, true, true, 8 >, [777](#)
- mulh\_fast
  - Simd128\_impl< true, true, false, 8 >, [629](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd256\_impl< true, true, false, 8 >, [710](#)
  - Simd256\_impl< true, true, true, 8 >, [749](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
- mulin
  - FieldSimd< \_Field >, [479](#)
  - ScalFunctions< Element >, [592](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
- mullo
  - ScalFunctionsBase< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, [597](#)
 >
    - Simd128\_impl< true, true, false, 2 >, [608](#)
    - Simd128\_impl< true, true, false, 4 >, [618](#)
    - Simd128\_impl< true, true, false, 8 >, [624](#)
    - Simd128\_impl< true, true, true, 2 >, [637](#)
    - Simd128\_impl< true, true, true, 4 >, [646](#)
    - Simd128\_impl< true, true, true, 8 >, [657](#)
    - Simd256\_impl< true, true, false, 2 >, [680](#)
    - Simd256\_impl< true, true, false, 4 >, [697](#)
    - Simd256\_impl< true, true, false, 8 >, [705](#)
    - Simd256\_impl< true, true, true, 2 >, [717](#)
    - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
    - Simd256\_impl< true, true, true, 8 >, [747](#)
    - Simd512\_impl< true, true, false, 8 >, [764](#)
    - Simd512\_impl< true, true, true, 8 >, [777](#)
- mulx
  - ScalFunctionsBase< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, [597](#)
 >
    - Simd128\_impl< true, true, false, 2 >, [604](#)
    - Simd128\_impl< true, true, false, 4 >, [614](#)
    - Simd128\_impl< true, true, false, 8 >, [625](#)
    - Simd128\_impl< true, true, true, 2 >, [637](#)
    - Simd128\_impl< true, true, true, 4 >, [647](#)
    - Simd128\_impl< true, true, true, 8 >, [657](#)
    - Simd256\_impl< true, true, false, 2 >, [676](#)
    - Simd256\_impl< true, true, false, 4 >, [688](#), [691](#)
    - Simd256\_impl< true, true, false, 8 >, [705](#)
    - Simd256\_impl< true, true, true, 2 >, [718](#)
    - Simd256\_impl< true, true, true, 4 >, [730](#), [737](#)
    - Simd256\_impl< true, true, true, 8 >, [747](#)
    - Simd512\_impl< true, true, false, 8 >, [764](#)
    - Simd512\_impl< true, true, true, 8 >, [777](#)
- mvcnt
  - test-lu.C, [1167](#)
- n
  - Bench< Elt >, [435](#)
  - count\_nonconst\_lvalue\_reference< const T &, O...
    - >, [463](#)
  - count\_nonconst\_lvalue\_reference< T &, O...
    - >, [463](#)
  - count\_nonconst\_lvalue\_reference< T, O...
    - >, [463](#)
  - count\_nonconst\_lvalue\_reference<>, [464](#)

- Sparse< \_Field, SparseMatrix\_t::COO >, 785
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 786
- Sparse< \_Field, SparseMatrix\_t::CSR >, 788
- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 791
- Sparse< \_Field, SparseMatrix\_t::ELL >, 793
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 794
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 796
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 798
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 800
- Sparse< \_Field, SparseMatrix\_t::SELL >, 801
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 803
- name
  - TestOneMethod< Simd >, 815
- nb\_lref
  - TestOneMethod< Simd >, 815
- nChunks
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 795
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 797
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 802
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 804
- nDenseCols
  - StatsMatrix, 808
- nDenseRows
  - StatsMatrix, 807
- need\_field\_characteristic< Field >, 552
  - value, 552
- need\_field\_characteristic< Givaro::Modular< Field > >, 552
  - value, 553
- need\_field\_characteristic< Givaro::ModularBalanced< Field > >, 553
  - value, 553
- NeedDoublePreAddReduction
  - FFLAS::Protected, 231
- NeedPreAddReduction
  - FFLAS::Protected, 230
- NeedPreAxyReduction
  - FFLAS::Protected, 234
- NeedPreScalReduction
  - FFLAS::Protected, 234
- NeedPreSubReduction
  - FFLAS::Protected, 231
- neg
  - RNSIntegerMod< RNS >, 586
- nElements
  - Sparse< \_Field, SparseMatrix\_t::COO >, 785
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 787
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 788
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 791
  - Sparse< \_Field, SparseMatrix\_t::ELL >, 793
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 795
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 796
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 798
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 800
- Sparse< \_Field, SparseMatrix\_t::SELL >, 802
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 804
- nEmptyCols
  - StatsMatrix, 808
- nEmptyColsEnd
  - StatsMatrix, 808
- nEmptyRows
  - StatsMatrix, 808
- newD
  - FFPACK::Protected, 422
- NEWWINO
  - fgemm\_winograd.inl, 1078
- nMOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
  - StatsMatrix, 806
- nnz
  - Sparse< \_Field, SparseMatrix\_t::COO >, 785
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 786
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 788
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 791
  - Sparse< \_Field, SparseMatrix\_t::ELL >, 793
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 795
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 796
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 798
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 800
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 802
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 804
  - StatsMatrix, 806
- none
  - HelperFlag, 503
- nOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790
  - StatsMatrix, 806
- nonsquare\_inplace\_v1
  - FFLAS::\_ftranspose\_impl, 204
- nonsquare\_inplace\_v2
  - FFLAS::\_ftranspose\_impl, 204
- NonZeroRandomMatrix
  - FFPACK, 401
- normA
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 536
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 539
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 542

normB  
   MMHelper< FFPACK::RNSInteger< E >, Algo-  
     Trait, ModeCategories::DefaultTag, ParSeq-  
     Trait >, 536  
   MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
     goTrait, ModeCategories::DefaultTag, ParSeq-  
     Trait >, 539  
   MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
     ElementCategories::RNSElementTag >, ParSeq-  
     Trait >, 542  
 NORML\_MOD  
   fflas\_simd.h, 985  
 NoSimd< T >, 553  
   aligned\_allocator, 554  
   aligned\_vector, 554  
   alignment, 554  
   compliant, 554  
   is\_same\_element, 554  
   scalar\_t, 554  
   type\_string, 554  
   valid, 554  
   vect\_size, 554  
   vect\_t, 553  
 NoSimdSparseMatrix  
   FFLAS, 77  
 NOSPLIT  
   parallel.h, 1100  
 not\_inplace  
   FFLAS::\_frtranspose\_impl, 203  
 nOthers  
   Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 790  
   StatsMatrix, 806  
 NotMKLSparseMatrixFormat  
   FFLAS, 77  
 NotZOSparseMatrix  
   FFLAS, 76  
 NullSpaceBasis  
   FFPACK, 349, 395  
 NullSpaceBasis\_modular\_double  
   ffpack.C, 1012  
   ffpack\_c.h, 1045  
 NUM\_THREADS  
   parallel.h, 1095  
 NUMARGS  
   parallel.h, 1098  
 number\_kind  
   FFLAS, 81  
 numBlock  
   ForStrategy1D< blocksize\_t, Cut, Param >, 492  
 numblocks  
   ForStrategy1D< blocksize\_t, Cut, Param >, 491  
 numColBlock  
   ForStrategy2D< blocksize\_t, Cut, Param >, 495  
 numRowBlock  
   ForStrategy2D< blocksize\_t, Cut, Param >, 495  
 numthreads  
   Parallel< C, P >, 555  
   Sequential, 599  
 one  
   FFLAS, 81  
   RNSInteger< RNS >, 581  
   RNSIntegerMod< RNS >, 588  
   Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 800  
 OPENBLAS\_NUM\_THREADS  
   config.h, 876  
 operator!=  
   rns\_double\_elt\_cstptr, 569  
   rns\_double\_elt\_ptr, 572  
 operator<  
   rns\_double\_elt\_cstptr, 569  
   rns\_double\_elt\_ptr, 572  
 operator<<  
   Compose< H1, H2 >, 450  
   FFLAS, 160  
   ForStrategy2D< blocksize\_t, Cut, Param >, 494  
   MMHelper< FFPACK::RNSInteger< E >, Algo-  
     Trait, ModeCategories::DefaultTag, ParSeq-  
     Trait >, 536  
   MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
     goTrait, ModeCategories::DefaultTag, ParSeq-  
     Trait >, 538  
   MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
     Dest >, ParSeqTrait >, 540  
   MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
     ElementCategories::RNSElementTag >, ParSeq-  
     Trait >, 542  
   MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
     ParSeqTrait >, 544  
   MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
     Trait >, 533  
   Parallel< C, P >, 555  
   Sequential, 600  
   test-fsytrf.C, 1151  
   test-simd.C, 1179  
 operator\*  
   rns\_double\_elt\_cstptr, 568  
   rns\_double\_elt\_ptr, 571  
 operator()  
   callLUdivine\_small< double >, 437  
   callLUdivine\_small< Element >, 437  
   callLUdivine\_small< float >, 438  
   Failure, 472, 473  
   readMyMachineType< Field, mpz\_t >, 559  
   readMyMachineType< Field, T >, 558  
   RNSInteger< RNS >::RandIter, 557  
   RNSIntegerMod< RNS >::RandIter, 558  
   rnsRandIter< RNS >, 589  
   ScaFunctionsBase< Element, typename enable\_if<  
     is\_floating\_point< Element >::value >::type  
     >::FloatingPointTestDistribution, 490  
   tfn\_minus, 816  
   tfn\_minus\_eq, 816  
   tfn\_mul, 817  
   tfn\_mul\_eq, 817  
   tfn\_plus, 817

- tfn\_plus\_eq, [818](#)
- operator+
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator++
  - ForStrategy1D< blocksize\_t, Cut, Param >, [491](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [493](#)
  - rns\_double\_elt\_cstptr, [568](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator+=
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator-
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator--
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator-=
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator=
  - Coo< Field >, [459](#)
  - Coo< ValT, IdxT >, [458](#), [461](#)
  - FieldSimd< \_Field >, [476](#)
  - Info, [508](#), [509](#)
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [572](#)
- operator&
  - rns\_double\_elt, [566](#)
  - rns\_double\_elt\_cstptr, [568](#), [569](#)
  - rns\_double\_elt\_ptr, [571](#), [572](#)
- operator[]
  - rns\_double\_elt\_cstptr, [568](#)
  - rns\_double\_elt\_ptr, [571](#)
- other
  - FFLAS, [81](#)
  - rns\_double\_elt\_cstptr, [569](#)
  - rns\_double\_elt\_ptr, [573](#)
- Outmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- Outmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [534](#)
- outputs\_scalar
  - TestOneMethod< Simd >, [816](#)
- outputs\_simd
  - TestOneMethod< Simd >, [816](#)
- p
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecisionTag >, [505](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecisionTag >, [506](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [506](#)
- pack
  - ScalFunctions< Element >, [594](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [695](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [735](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [756](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- pack\_even
  - ScalFunctions< Element >, [594](#)
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [694](#), [695](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [735](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [766](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- pack\_lhs
  - FFLAS::details, [219](#)
- pack\_odd
  - ScalFunctions< Element >, [594](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [695](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [735](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- pack\_rhs
  - FFLAS::details, [220](#)

PACKAGE  
     config.h, [876](#)  
 PACKAGE\_BUGREPORT  
     config.h, [876](#)  
 PACKAGE\_NAME  
     config.h, [876](#)  
 PACKAGE\_STRING  
     config.h, [876](#)  
 PACKAGE\_TARNAME  
     config.h, [876](#)  
 PACKAGE\_URL  
     config.h, [876](#)  
 PACKAGE\_VERSION  
     config.h, [876](#)  
 PAR\_BLOCK  
     parallel.h, [1095](#)  
 Parallel  
     Parallel< C, P >, [555](#)  
 Parallel< C, P >, [555](#)  
     Cut, [555](#)  
     numthreads, [555](#)  
     operator<<, [555](#)  
     Parallel, [555](#)  
     Param, [555](#)  
     set\_numthreads, [555](#)  
 parallel.h, [1094](#)  
     \_\_FFLASFFPACK\_SEQUENTIAL, [1095](#)  
     BARRIER, [1095](#)  
     BEGIN\_PARALLEL\_MAIN, [1096](#)  
     CHECK\_DEPENDENCIES, [1095](#)  
     COMMA, [1098](#)  
     CONSTREFERENCE, [1096](#)  
     END\_PARALLEL\_MAIN, [1096](#)  
     FOR1D, [1097](#)  
     FOR2D, [1097](#)  
     FORBLOCK1D, [1096](#)  
     FORBLOCK2D, [1097](#)  
     index\_t, [1095](#)  
     MAX\_THREADS, [1096](#)  
     MODE, [1098](#)  
     NOSPLIT, [1100](#)  
     NUM\_THREADS, [1095](#)  
     NUMARGS, [1098](#)  
     PAR\_BLOCK, [1095](#)  
     PARFOR1D, [1097](#)  
     PARFOR2D, [1098](#)  
     PARFORBLOCK1D, [1097](#)  
     PARFORBLOCK2D, [1098](#)  
     PP\_ARG\_N, [1099](#)  
     PP\_NARG\_, [1098](#)  
     PP\_RSEQ\_N, [1100](#)  
     READ, [1096](#)  
     READWRITE, [1096](#)  
     RETURNPARAM, [1098](#)  
     SET\_THREADS, [1096](#)  
     splitt, [1100](#)  
     SPLITTER, [1101](#)  
     splitting\_0, [1100](#)  
     splitting\_1, [1100](#)  
     splitting\_2, [1100](#)  
     splitting\_3, [1100](#)  
     SYNCH\_GROUP, [1095](#)  
     TASK, [1095](#)  
     THREAD\_INDEX, [1095](#)  
     VALUE, [1096](#)  
     WAIT, [1095](#)  
     WRITE, [1096](#)  
 Param  
     Parallel< C, P >, [555](#)  
 PARFOR1D  
     parallel.h, [1097](#)  
 PARFOR2D  
     parallel.h, [1098](#)  
 PARFORBLOCK1D  
     parallel.h, [1097](#)  
 PARFORBLOCK2D  
     parallel.h, [1098](#)  
 parseArguments  
     FFLAS, [198](#)  
 parseq  
     MMHelper< FFPACK::RNSInteger< E >, Algo-  
         Trait, ModeCategories::DefaultTag, ParSeq-  
         Trait >, [537](#)  
     MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
         goTrait, ModeCategories::DefaultTag, ParSeq-  
         Trait >, [539](#)  
     MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
         Dest >, ParSeqTrait >, [540](#)  
     MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
         ElementCategories::RNSElementTag >,  
         ParSeqTrait >, [543](#)  
     MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
         ParSeqTrait >, [544](#)  
     MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
         Trait >, [534](#)  
     TRSMHelper< ReclterTrait, ParSeqTrait >, [820](#)  
 PBASECASE\_K  
     ffpack\_ppluq.inl, [1073](#)  
 pColumnEchelonForm  
     FFPACK, [336](#)  
 pColumnEchelonForm\_modular\_double  
     ffpack.C, [1006](#)  
 pColumnEchelonForm\_modular\_float  
     ffpack.C, [1007](#)  
 pColumnEchelonForm\_modular\_int32\_t  
     ffpack.C, [1008](#)  
 pColumnRankProfile  
     FFPACK, [352](#)  
 pDet  
     FFPACK, [347](#)  
 perm  
     Info, [508](#), [510](#)  
     Sparse< \_Field, SparseMatrix\_t::SELL >, [802](#)  
     Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)  
 PermApplyS  
     FFPACK, [379](#)

- PermApplyS\_double
  - ffpack.C, [999](#)
  - ffpack\_c.h, [1034](#)
- PermApplyT
  - FFPACK, [381](#)
- PermApplyT\_double
  - ffpack.C, [999](#)
  - ffpack\_c.h, [1034](#)
- pfadd
  - FFLAS, [84](#)
- pfaddin
  - FFLAS, [85](#)
- pfgemm
  - FFLAS, [151](#), [194–196](#)
- pfgemm\_1D\_rec
  - FFLAS, [152](#)
- pfgemm\_2D\_rec
  - FFLAS, [152](#)
- pfgemm\_3D\_rec
  - FFLAS, [152](#)
- pfgemm\_3D\_rec2
  - FFLAS, [153](#)
- pfgemm\_variants.inl, [1101](#)
- pfgemv.inl, [1102](#)
- pfrand
  - FFLAS, [194](#)
- pfreduce
  - FFLAS, [116](#)
- pfspmm
  - FFLAS::sparse\_details, [249–251](#)
  - FFLAS::sparse\_details\_impl, [267](#), [274–276](#), [278](#), [279](#), [289](#), [290](#)
- pfspmm\_dispatch
  - FFLAS::sparse\_details, [248](#)
- pfspmm\_mone
  - FFLAS::sparse\_details\_impl, [268](#)
- pfspmm\_one
  - FFLAS::sparse\_details\_impl, [267](#), [268](#)
- pfspmm\_zo
  - FFLAS::sparse\_details\_impl, [279](#), [280](#)
- pfspmv
  - FFLAS::sparse\_details, [252](#), [253](#)
  - FFLAS::sparse\_details\_impl, [268](#), [269](#), [276](#), [280](#), [286](#), [290](#), [292](#)
- pfspmv\_mone
  - FFLAS::sparse\_details\_impl, [269](#), [270](#), [281](#), [286](#), [287](#), [293](#)
- pfspmv\_one
  - FFLAS::sparse\_details\_impl, [269](#), [270](#), [281](#), [286](#), [287](#), [293](#)
- pfspmv\_task
  - FFLAS::sparse\_details\_impl, [269](#)
- pfsub
  - FFLAS, [85](#)
- pfsubin
  - FFLAS, [85](#)
- pfzero
  - FFLAS, [194](#)
- PLUQ
  - FFPACK, [333](#), [334](#), [384](#), [385](#), [389](#)
- pluq.C, [1102](#), [1103](#)
- PLUQ\_basecaseCROUT
  - FFPACK, [384](#)
- PLUQ\_basecaseV2
  - FFPACK, [384](#)
- PLUQ\_basecaseV3
  - FFPACK, [383](#)
- PLUQ\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1038](#)
- PLUQtoEchelonPermutation
  - FFPACK, [362](#)
  - ffpack.C, [1016](#)
  - ffpack\_c.h, [1049](#)
- pm1
  - HelperFlag, [503](#)
- pMMH
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [819](#), [820](#)
- pow50rem
  - HelperMod< Field, ElementCategories::MachineIntTag >, [504](#)
- PP\_ARG\_N
  - parallel.h, [1099](#)
- PP\_NARG\_
  - parallel.h, [1098](#)
- PP\_RSEQ\_N
  - parallel.h, [1100](#)
- pPLUQ
  - FFPACK, [333](#)
- pRank
  - FFPACK, [345](#)
- preamble
  - FFLAS, [199](#)
- precompute\_cst
  - rns\_double, [562](#)
  - rns\_double\_extended, [575](#)
- pReducedColumnEchelonForm
  - FFPACK, [338](#)
- pReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [1007](#)
- pReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [1008](#)
- pReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [1009](#)
- pReducedRowEchelonForm
  - FFPACK, [339](#)
- pReducedRowEchelonForm\_modular\_double
  - ffpack.C, [1007](#)
- pReducedRowEchelonForm\_modular\_float
  - ffpack.C, [1008](#)
- pReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [1009](#)
- prefetch
  - FFLAS, [202](#)
- print

- Failure, [473](#)
- printHelpMessage
  - args-parser.h, [826](#)
- printPolynomial
  - test-charpoly-check.C, [1128](#)
- printvect
  - test-compressQ.C, [1130](#)
- productBruhatxTS
  - FFPACK, [367](#), [370](#)
- pRowEchelonForm
  - FFPACK, [337](#)
- pRowEchelonForm\_modular\_double
  - ffpack.C, [1006](#)
- pRowEchelonForm\_modular\_float
  - ffpack.C, [1007](#)
- pRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [1008](#)
- pRowRankProfile
  - FFPACK, [351](#)
- PSeq
  - benchmark-fgemm-rns.C, [839](#)
- pSolve
  - FFPACK, [349](#)
- PTRSM\_HYBRID\_THRESHOLD
  - fflas\_ptrsm.inl, [981](#)
- PURE
  - fflas\_simd.h, [985](#)
- queryCacheSizes
  - FFLAS, [202](#)
- queryL1CacheSize
  - FFLAS, [202](#)
- queryTopLevelCacheSize
  - FFLAS, [202](#)
- RandInt
  - FFPACK, [404](#)
- RandIter
  - RNSInteger< RNS >::RandIter, [556](#)
  - RNSIntegerMod< RNS >::RandIter, [557](#)
- random
  - RNSInteger< RNS >::RandIter, [556](#)
  - RNSIntegerMod< RNS >::RandIter, [557](#), [558](#)
  - rnsRandIter< RNS >, [589](#)
- RandomIndexSubset
  - FFPACK, [406](#)
- RandomKrylovPrecond
  - FFPACK::Protected, [421](#)
- RandomLTQSMMatrixWithRankandQSorder
  - FFPACK, [416](#)
- RandomLTQSRankProfileMatrix
  - FFPACK, [408](#)
- RandomMatrix
  - FFPACK, [402](#)
- RandomMatrixWithDet
  - FFPACK, [414](#)
- RandomMatrixWithRank
  - FFPACK, [405](#), [406](#)
- RandomMatrixWithRankandRandomRPM
  - FFPACK, [411](#)
- RandomMatrixWithRankandRPM
  - FFPACK, [408](#), [409](#)
- RandomNullSpaceVector
  - FFPACK, [349](#), [368](#), [395](#)
- RandomNullSpaceVector\_modular\_double
  - ffpack.C, [1012](#)
  - ffpack\_c.h, [1045](#)
- RandomPermutation
  - FFPACK, [407](#)
- RandomRankProfileMatrix
  - FFPACK, [407](#)
- RandomSymmetricMatrix
  - FFPACK, [404](#)
- RandomSymmetricMatrixWithRankandRandomRPM
  - FFPACK, [412](#)
- RandomSymmetricMatrixWithRankandRPM
  - FFPACK, [409](#), [410](#)
- RandomSymmetricRankProfileMatrix
  - FFPACK, [407](#)
- RandomTriangularMatrix
  - FFPACK, [403](#), [404](#)
- Rank
  - FFPACK, [345](#), [346](#), [393](#)
- rank.C, [1104](#)
  - main, [1104](#)
- Rank\_modular\_double
  - ffpack.C, [1010](#)
  - ffpack\_c.h, [1043](#)
- RankProfileFromLU
  - FFPACK, [352](#)
  - ffpack.C, [1013](#)
  - ffpack\_c.h, [1046](#)
- READ
  - parallel.h, [1096](#)
- read\_field
  - Matio.h, [1093](#)
- read\_sparse.h, [1104](#)
  - DNS\_BIN\_VER, [1105](#)
  - mask\_t, [1105](#)
- readDnsFormat
  - FFLAS, [161](#)
- readMachineType
  - FFLAS, [161](#)
- ReadMatrix
  - FFLAS, [199](#), [200](#)
- readMyMachineType< Field, mpz\_t >, [559](#)
  - Element, [559](#)
  - Element\_ptr, [559](#)
  - operator(), [559](#)
- readMyMachineType< Field, T >, [558](#)
  - Element, [558](#)
  - Element\_ptr, [558](#)
  - operator(), [558](#)
- readOrRandomMatrixWithRankAndRandomRPM
  - test-nullspace.C, [1170](#)
- readSmsFormat
  - FFLAS, [160](#)

- readSprFormat
  - FFLAS, [160](#)
- READWRITE
  - parallel.h, [1096](#)
- Rec\_Initialize
  - benchmark-pluq.C, [853](#)
- RecInt, [424](#)
- recLevel
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [537](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [539](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConversionTag, Dest >, ParSeqTrait >, [540](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConversionTag, ElementCategories::RNSElementTag >, ParSeqTrait >, [542](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [544](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [533](#)
- Recursive, [560](#)
- reduce
  - FFLAS::vectorised, [300](#), [301](#)
  - rns\_double, [563](#)
  - rns\_double\_extended, [576](#)
  - RNSInteger< RNS >, [580](#)
  - RNSIntegerMod< RNS >, [585](#)
- reduce\_modp
  - RNSIntegerMod< RNS >, [586](#), [587](#)
- reduce\_modp\_rnsmajor
  - RNSIntegerMod< RNS >, [587](#)
- ReducedColumnEchelonForm
  - FFPACK, [337](#), [338](#), [390](#)
- ReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [1004](#)
  - ffpack\_c.h, [1040](#)
- ReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [1005](#)
  - ffpack\_c.h, [1041](#)
- ReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1041](#)
- ReducedRowEchelonForm
  - FFPACK, [339](#), [340](#), [390](#)
- ReducedRowEchelonForm2\_modular\_double
  - ffpack\_c.h, [1042](#)
- ReducedRowEchelonForm\_modular\_double
  - ffpack.C, [1004](#)
  - ffpack\_c.h, [1040](#)
- ReducedRowEchelonForm\_modular\_float
  - ffpack.C, [1005](#)
  - ffpack\_c.h, [1041](#)
- ReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1041](#)
- REF\_modular\_double
  - ffpack\_c.h, [1042](#)
- regression-check.C, [1105](#)
  - check1, [1105](#)
  - check2, [1105](#)
  - check3, [1106](#)
  - check4, [1106](#)
  - checkZeroDimCharpoly, [1106](#)
  - checkZeroDimMinPoly, [1106](#)
  - gf2ModularBalanced, [1106](#)
  - main, [1106](#)
- Residu
  - Bench< Elt >, [433](#)
  - Test< Elt >, [811](#)
- RETURNPARAM
  - parallel.h, [1098](#)
- Ring
  - CheckerImplem\_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, [440](#)
- RNSInteger< RNS >::RandIter, [557](#)
- RNSIntegerMod< RNS >::RandIter, [558](#)
- rnsRandIter< RNS >, [589](#)
- rint< K >, [560](#)
- RNS, [47](#)
  - benchmark-fgemm-rns.C, [838](#)
- rns
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [583](#)
- rns-double-elt.h, [1106](#)
- rns-double-recint.inl, [1107](#)
  - \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL, [1107](#)
- rns-double.h, [1107](#)
  - ROUND\_DOWN, [1108](#)
- rns-double.inl, [1108](#)
  - \_\_FFLASFFPACK\_field\_rns\_double\_INL, [1108](#)
- rns-integer-mod.h, [1108](#)
- rns-integer.h, [1109](#)
- rns.h, [1110](#)
- rns.inl, [1110](#)
  - \_\_FFLASFFPACK\_field\_rns\_INL, [1110](#)
- rns\_double, [560](#)
  - \_M, [564](#)
  - \_MMi, [565](#)
  - \_Mi, [564](#)
  - \_basis, [564](#)
  - \_basisMax, [564](#)
  - \_crt\_in, [565](#)
  - \_crt\_out, [565](#)
  - \_field\_rns, [564](#)
  - \_invbasis, [564](#)
  - \_ldm, [565](#)
  - \_mi\_sum, [565](#)
  - \_negbasis, [564](#)
  - \_pbits, [565](#)
  - \_size, [565](#)
- BasisElement, [561](#)

- ConstElement\_ptr, 561
- convert, 563, 564
- convert\_transpose, 563
- Element, 561
- Element\_ptr, 561
- init, 562, 563
- init\_transpose, 563
- integer, 561
- ModField, 561
- precompute\_cst, 562
- reduce, 563
- rns\_double, 561, 562
- rns\_double\_elt, 565
  - \_alloc, 567
  - \_ptr, 566
  - \_stride, 567
  - ~rns\_double\_elt, 566
  - operator&, 566
  - rns\_double\_elt, 566
- rns\_double\_elt\_cstptr, 567
  - \_alloc, 570
  - \_ptr, 570
  - \_stride, 570
  - operator!=, 569
  - operator<, 569
  - operator\*, 568
  - operator+, 569
  - operator++, 568
  - operator+=, 569
  - operator-, 569
  - operator--, 569
  - operator-=, 569
  - operator=, 569
  - operator&, 568, 569
  - operator[], 568
  - other, 569
  - rns\_double\_elt\_cstptr, 568
- rns\_double\_elt\_ptr, 570
  - \_alloc, 573
  - \_ptr, 573
  - \_stride, 573
  - operator!=, 572
  - operator<, 572
  - operator\*, 571
  - operator+, 572
  - operator++, 572
  - operator+=, 572
  - operator-, 572
  - operator--, 572
  - operator-=, 572
  - operator=, 572
  - operator&, 571, 572
  - operator[], 571
  - other, 573
  - rns\_double\_elt\_ptr, 571
- rns\_double\_extended, 573
  - \_M, 577
  - \_MMi, 577
  - \_Mi, 577
  - \_basis, 576
  - \_basisMax, 576
  - \_crt\_in, 577
  - \_crt\_out, 577
  - \_field\_rns, 577
  - \_invbasis, 576
  - \_ldm, 577
  - \_negbasis, 576
  - \_pbits, 577
  - \_size, 577
  - BasisElement, 574
  - ConstElement\_ptr, 574
  - convert, 575, 576
  - Element, 574
  - Element\_ptr, 574
  - init, 575, 576
  - integer, 574
  - ModField, 574
  - precompute\_cst, 575
  - reduce, 576
  - rns\_double\_extended, 574, 575
- RNSElementTag, 577
- RNSInteger
  - RNSInteger< RNS >, 579
- RNSInteger< RNS >, 578
  - \_rns, 581
  - assign, 581
  - BasisElement, 579
  - cardinality, 580
  - characteristic, 580
  - ConstElement\_ptr, 579
  - convert, 580
  - Element, 579
  - Element\_ptr, 579
  - init, 580
  - integer, 579
  - isMOne, 580
  - isOne, 579
  - isZero, 580
  - mOne, 581
  - one, 581
  - reduce, 580
  - rns, 579
  - RNSInteger, 579
  - size, 579
  - write, 581
  - zero, 581
- RNSInteger< RNS >::RandIter, 556
  - operator(), 557
  - RandIter, 556
  - random, 556
  - ring, 557
- RNSIntegerMod
  - RNSIntegerMod< RNS >, 583
- RNSIntegerMod< RNS >, 581
  - \_F, 587
  - \_Mi\_modp\_rns, 587

- `_RNSdelayed`, 588
- `_iM_modp_rns`, 587
- `_p`, 587
- `_rns`, 587
- `add`, 585
- `areEqual`, 586
- `assign`, 585
- `axpyin`, 586
- `BasisElement`, 583
- `cardinality`, 584
- `characteristic`, 584
- `ConstElement_ptr`, 583
- `convert`, 585
- `delayed`, 583
- `Element`, 583
- `Element_ptr`, 583
- `init`, 584, 585
- `integer`, 583
- `inv`, 586
- `isMOne`, 584
- `isOne`, 584
- `isZero`, 584
- `maxElement`, 584
- `minElement`, 584
- `ModField`, 583
- `mOne`, 588
- `mul`, 586
- `neg`, 586
- `one`, 588
- `reduce`, 585
- `reduce_modp`, 586, 587
- `reduce_modp_rnsmajor`, 587
- `rns`, 583
- `RNSIntegerMod`, 583
- `size`, 584
- `sub`, 585
- `write`, 586
- `write_matrix`, 586
- `write_matrix_long`, 587
- `zero`, 588
- `RNSIntegerMod< RNS >::RandIter`, 557
  - `operator()`, 558
  - `RandIter`, 557
  - `random`, 557, 558
  - `ring`, 558
- `RNSModulus`
  - `FFLAS::CuttingStrategy`, 211
- `rnsRandIter`
  - `rnsRandIter< RNS >`, 588
- `rnsRandIter< RNS >`, 588
  - `operator()`, 589
  - `random`, 589
  - `ring`, 589
  - `rnsRandIter`, 588
- `round`
  - `ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >`, 595
  - `ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >`, 597
  - `Simd128_impl< true, true, false, 2 >`, 609
  - `Simd128_impl< true, true, false, 4 >`, 619
  - `Simd128_impl< true, true, false, 8 >`, 629
  - `Simd128_impl< true, true, true, 2 >`, 639
  - `Simd128_impl< true, true, true, 4 >`, 649
  - `Simd128_impl< true, true, true, 8 >`, 659
  - `Simd256_impl< true, false, true, 8 >`, 671
  - `Simd256_impl< true, true, false, 2 >`, 681
  - `Simd256_impl< true, true, false, 4 >`, 699
  - `Simd256_impl< true, true, false, 8 >`, 710
  - `Simd256_impl< true, true, true, 2 >`, 720
  - `Simd256_impl< true, true, true, 4 >`, 732, 739
  - `Simd256_impl< true, true, true, 8 >`, 749
  - `Simd512_impl< true, false, true, 8 >`, 759
  - `Simd512_impl< true, true, false, 8 >`, 769
  - `Simd512_impl< true, true, true, 8 >`, 779
- `ROUND_DOWN`
  - `fflas_sparse.h`, 990
  - `rns-double.h`, 1108
- `Row`, 589
- `row`
  - `Coo< Field >`, 460
  - `Coo< ValT, IdxT >`, 458, 461
  - `Sparse< _Field, SparseMatrix_t::COO >`, 784
  - `Sparse< _Field, SparseMatrix_t::COO_ZO >`, 786
- `rowblockindex`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, 494
- `rowBlockSize`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, 494
- `rowdim`
  - `StatsMatrix`, 806
- `RowEchelonForm`
  - `FFPACK`, 336, 337, 390
- `RowEchelonForm_modular_double`
  - `ffpack.C`, 1003
  - `ffpack_c.h`, 1039
- `RowEchelonForm_modular_float`
  - `ffpack.C`, 1004
  - `ffpack_c.h`, 1039
- `RowEchelonForm_modular_int32_t`
  - `ffpack.C`, 1005
  - `ffpack_c.h`, 1040
- `rownumblocks`
  - `ForStrategy2D< blocksize_t, Cut, Param >`, 493
- `RowRankProfile`
  - `FFPACK`, 350, 351, 396
- `RowRankProfile_modular_double`
  - `ffpack.C`, 1012
  - `ffpack_c.h`, 1045
- `RowRankProfileSubmatrix`
  - `FFPACK`, 355, 397
- `RowRankProfileSubmatrix_modular_double`
  - `ffpack.C`, 1013
  - `ffpack_c.h`, 1046
- `RowRankProfileSubmatrixIndices`
  - `FFPACK`, 353, 396

- RowRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [1013](#)
  - ffpack\_c.h, [1046](#)
- ruint< K >, [589](#)
- run
  - Bench< Elt >, [434](#)
  - Test< Elt >, [812](#)
- run\_with\_field
  - benchmark-charpoly.C, [829](#)
  - benchmark-fdot.C, [837](#)
  - benchmark-quasisep.C, [854](#)
  - test-charpoly.C, [1129](#)
  - test-echelon.C, [1133](#)
  - test-fdot.C, [1136](#)
  - test-fgemm-check.C, [1137](#)
  - test-fgemm.C, [1139](#)
  - test-fgemv.C, [1141](#)
  - test-fger.C, [1143](#)
  - test-fgesv.C, [1144](#)
  - test-finit.C, [1145](#)
  - test-fsyr2k.C, [1148](#)
  - test-fsyrk.C, [1150](#)
  - test-fsytrf.C, [1152](#)
  - test-ftrmm.C, [1153](#)
  - test-ftrmv.C, [1154](#)
  - test-ftrsm.C, [1156](#)
  - test-ftrssyr2k.C, [1157](#)
  - test-ftrstr.C, [1159](#)
  - test-ftrsv.C, [1160](#)
  - test-ftrtri.C, [1161](#)
  - test-io.C, [1163](#)
  - test-lu.C, [1166](#)
  - test-minpoly.C, [1169](#)
  - test-nullspace.C, [1171](#)
  - test-quasisep.C, [1174](#)
  - test-rankprofiles.C, [1175](#)
  - test-solve.C, [1180](#)
- run\_with\_Integer
  - test-fdot.C, [1136](#)
- saxpy\_
  - config-blas.h, [869](#)
- ScalAndReduce
  - FFLAS::Protected, [232](#), [233](#)
- scalar\_t
  - FieldSimd< \_Field >, [476](#)
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [612](#)
  - Simd128\_impl< true, true, false, 8 >, [622](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [685](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [724](#), [725](#)
  - Simd256\_impl< true, true, true, 8 >, [743](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [761](#)
  - Simd512\_impl< true, true, true, 8 >, [772](#)
- ScalFunctions< Element >, [589](#)
  - add, [591](#)
  - addin, [591](#)
  - blend, [594](#)
  - div, [592](#)
  - eq, [593](#)
  - fmadd, [592](#)
  - fmaddin, [592](#)
  - fmsub, [592](#)
  - fmsubin, [592](#)
  - fnmadd, [592](#)
  - fnmaddin, [593](#)
  - genInputs, [590](#)
  - genInputsWithZero, [590](#)
  - greater, [593](#)
  - greater\_eq, [593](#)
  - lesser, [593](#)
  - lesser\_eq, [593](#)
  - mul, [592](#)
  - mulin, [592](#)
  - pack, [594](#)
  - pack\_even, [594](#)
  - pack\_odd, [594](#)
  - sub, [591](#)
  - subin, [591](#)
  - unpackhi, [593](#)
  - unpacklo, [593](#)
  - unpacklohi, [594](#)
  - vand, [591](#)
  - vandnot, [591](#)
  - vectElt, [590](#)
  - vor, [591](#)
  - vxor, [591](#)
  - zero, [591](#)
- ScalFunctionsBase< Element, Enable >, [594](#)
- ScalFunctionsBase< Element, typename enable\_if<
  - is\_floating\_point< Element >::value >::type
  - >, [595](#)
  - \_zero, [596](#)
  - blendv, [595](#)
  - ceil, [595](#)
  - cmp\_false, [596](#)
  - cmp\_true, [596](#)
  - floor, [595](#)
  - fma, [596](#)
  - get\_default\_random\_generator, [595](#)
  - round, [595](#)
- ScalFunctionsBase< Element, typename enable\_if<
  - is\_floating\_point< Element >::value >::type
  - >::FloatingPointTestDistribution, [489](#)
  - FloatingPointTestDistribution, [489](#)
  - IntType, [489](#)
  - operator(), [490](#)

- ScalFunctionsBase< Element, typename enable\_if<  
is\_integral< Element >::value >::type >, 596
- \_zero, 598
- cmp\_false, 599
- cmp\_true, 599
- fma, 597
- fmaddx, 597
- fmaddxin, 597
- fmsubx, 598
- fmsubxin, 598
- fnmaddx, 598
- fnmaddxin, 598
- get\_default\_random\_generator, 597
- mulhi, 597
- mullo, 597
- mulx, 597
- round, 597
- sll, 598
- sra, 598
- srl, 598
- scalp
  - FFLAS::vectorised, 302
  - FFLAS::vectorised::unswitch, 304, 305
- schedule\_bini.inl, 1110
  - \_\_FFLASFFPACK\_fgemm\_bini\_INL, 1111
- schedule\_winograd.inl, 1111
  - \_\_FFLASFFPACK\_fgemm\_winograd\_INL, 1111
- schedule\_winograd\_acc.inl, 1112
  - \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL, 1112
- schedule\_winograd\_acc\_ip.inl, 1112
  - \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL, 1113
- schedule\_winograd\_ip.inl, 1113
  - \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL, 1114
- scopy\_
  - config-blas.h, 871
- sdot\_
  - config-blas.h, 869
- second\_component
  - Compose< H1, H2 >, 450
- Self
  - Coo< ValT, IdxT >, 457, 460
- Self\_t
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 535
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 537
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Simd256\_impl< true, true, true, 2 >, 714
  - Dest >, ParSeqTrait >, 539
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Simd256\_impl< true, true, true, 8 >, 743
  - ElementCategories::RNSElementTag >, Simd512\_impl< true, false, true, 8 >, 754
  - ParSeqTrait >, 541
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, Simd512\_impl< true, true, true, 8 >, 773
  - ParSeqTrait >, 543
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 531
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 799
- SELL
  - FFLAS, 82
- sell.h, 1114
- sell\_pspmv.inl, 1114
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL, 1115
- sell\_spmv.inl, 1115
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL, 1116
- sell\_utils.inl, 1116
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL, 1116
- SELL\_ZO
  - FFLAS, 82
- Sequential, 599
  - numthreads, 599
  - operator<<, 600
  - Sequential, 599
- set
  - Simd128\_impl< true, true, false, 2 >, 603
  - Simd128\_impl< true, true, false, 4 >, 613
  - Simd128\_impl< true, true, false, 8 >, 623
  - Simd128\_impl< true, true, true, 2 >, 633
  - Simd128\_impl< true, true, true, 4 >, 643
  - Simd128\_impl< true, true, true, 8 >, 654
  - Simd256\_impl< true, false, true, 8 >, 666
  - Simd256\_impl< true, true, false, 2 >, 674
  - Simd256\_impl< true, true, false, 4 >, 686, 689, 692
  - Simd256\_impl< true, true, false, 8 >, 704
  - Simd256\_impl< true, true, true, 2 >, 714
  - Simd256\_impl< true, true, true, 4 >, 726, 733
  - Simd256\_impl< true, true, true, 8 >, 743
  - Simd512\_impl< true, false, true, 8 >, 754
  - Simd512\_impl< true, true, false, 8 >, 762, 765
  - Simd512\_impl< true, true, true, 8 >, 773
- set1
  - Simd128\_impl< true, true, false, 2 >, 603
  - Simd128\_impl< true, true, false, 4 >, 613
  - Simd128\_impl< true, true, false, 8 >, 623
  - Simd128\_impl< true, true, true, 2 >, 633
  - Simd128\_impl< true, true, true, 4 >, 643
  - Simd128\_impl< true, true, true, 8 >, 653
  - Simd256\_impl< true, false, true, 8 >, 665
  - Simd256\_impl< true, true, false, 2 >, 674
  - Simd256\_impl< true, true, false, 4 >, 686, 689
  - Simd256\_impl< true, true, false, 8 >, 704
  - Simd256\_impl< true, true, true, 2 >, 714
  - Simd256\_impl< true, true, true, 4 >, 726, 733
  - Simd512\_impl< true, false, true, 8 >, 754
  - Simd512\_impl< true, true, false, 8 >, 762
- set\_numthreads
  - Parallel< C, P >, 555

- SET\_THREADS
  - parallel.h, [1096](#)
- setErrorStream
  - Failure, [473](#)
- setNorm
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [536](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [538](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [542](#)
- setOutBounds
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [532](#)
- sgemm\_
  - config-blas.h, [872](#)
- sgemv\_
  - config-blas.h, [870](#)
- sger\_
  - config-blas.h, [870](#)
- shuffle
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [693](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [766](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- shuffle\_twice
  - Simd256\_impl< true, true, false, 4 >, [693](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
- sigma
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [801](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- signbits
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd256\_impl< true, true, false, 8 >, [710](#)
  - Simd256\_impl< true, true, true, 8 >, [750](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
- Simd
  - fflas\_simd.h, [986](#)
- simd
  - FieldSimd< \_Field >, [475](#)
- SIMD wrapper, [46](#)
- simd.doxy, [1116](#)
- Simd128
  - simd128.inl, [1117](#)
- simd128.inl, [1116](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL, [1117](#)
  - Simd128, [1117](#)
  - simd128\_double.inl, [1117](#)
    - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL, [1117](#)
  - simd128\_float.inl, [1118](#)
    - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL, [1118](#)
  - Simd128\_impl< ArithType, Int, Signed, Size >, [600](#)
  - Simd128\_impl< true, false, true, 4 >, [600](#)
  - Simd128\_impl< true, false, true, 8 >, [600](#)
  - Simd128\_impl< true, true, false, 2 >, [600](#)
  - add, [607](#)
  - addin, [607](#)
  - aligned\_allocator, [602](#)
  - aligned\_vector, [602](#)
  - alignment, [610](#)
  - blend, [607](#)
  - compliant, [605](#)
  - eq, [609](#)
  - fmadd, [608](#)
  - fmaddin, [608](#)
  - fmaddx, [604](#)
  - fmaddxin, [605](#)
  - fmsub, [608](#)
  - fmsubin, [609](#)
  - fmsubx, [605](#)
  - fmsubxin, [605](#)
  - fnmadd, [608](#)
  - fnmaddin, [608](#)
  - fnmaddx, [605](#)
  - fnmaddxin, [605](#)
  - gather, [603](#)
  - greater, [604](#)
  - greater\_eq, [604](#)
  - hadd\_to\_scal, [605](#)
  - is\_same\_element, [602](#)
  - lesser, [604](#)
  - lesser\_eq, [604](#)
  - load, [603](#)
  - loadu, [603](#)
  - mod, [609](#)
  - mul, [608](#)
  - mulhi, [604](#)
  - mullo, [608](#)
  - mulx, [604](#)
  - pack, [607](#)
  - pack\_even, [606](#)
  - pack\_odd, [607](#)
  - round, [609](#)
  - scalar\_t, [602](#)
  - set, [603](#)
  - set1, [603](#)
  - shuffle, [606](#)

- sll, [606](#)
- sll128, [609](#)
- sra, [604](#)
- srl, [606](#)
- srl128, [609](#)
- store, [603](#)
- storeu, [603](#)
- stream, [603](#)
- sub, [607](#)
- subin, [608](#)
- transpose, [607](#)
- type\_string, [602](#)
- unpackhi, [606](#)
- unpackhi\_intrinsic, [606](#)
- unpacklo, [606](#)
- unpacklo\_intrinsic, [606](#)
- unpacklohi, [606](#)
- valid, [605](#)
- vand, [609](#)
- vandnot, [610](#)
- vect\_size, [610](#)
- vect\_t, [602](#)
- vor, [610](#)
- vxor, [610](#)
- zero, [609](#)
- Simd128\_impl< true, true, false, 2 >::Converter, [451](#)
- t, [451](#)
- v, [451](#)
- Simd128\_impl< true, true, false, 4 >, [610](#)
- add, [617](#)
- addin, [617](#)
- aligned\_allocator, [612](#)
- aligned\_vector, [612](#)
- alignment, [620](#)
- blend, [617](#)
- compliant, [615](#)
- eq, [619](#)
- fmadd, [618](#)
- fmaddin, [618](#)
- fmaddx, [614](#)
- fmaddxin, [615](#)
- fmsub, [618](#)
- fmsubin, [618](#)
- fmsubx, [615](#)
- fmsubxin, [615](#)
- fnmadd, [618](#)
- fnmaddin, [618](#)
- fnmaddx, [615](#)
- fnmaddxin, [615](#)
- gather, [613](#)
- greater, [614](#)
- greater\_eq, [614](#)
- hadd\_to\_scal, [615](#)
- is\_same\_element, [612](#)
- lesser, [614](#)
- lesser\_eq, [614](#)
- load, [613](#)
- loadu, [613](#)
- mod, [619](#)
- mul, [618](#)
- mulhi, [614](#)
- mullo, [618](#)
- mulx, [614](#)
- pack, [617](#)
- pack\_even, [616](#)
- pack\_odd, [617](#)
- round, [619](#)
- scalar\_t, [612](#)
- set, [613](#)
- set1, [613](#)
- shuffle, [616](#)
- sll, [616](#)
- sll128, [619](#)
- sra, [614](#)
- srl, [616](#)
- srl128, [619](#)
- store, [613](#)
- storeu, [613](#)
- stream, [613](#)
- sub, [617](#)
- subin, [617](#)
- transpose, [617](#)
- type\_string, [613](#)
- unpackhi, [616](#)
- unpackhi\_intrinsic, [616](#)
- unpacklo, [616](#)
- unpacklo\_intrinsic, [616](#)
- unpacklohi, [616](#)
- valid, [615](#)
- vand, [619](#)
- vandnot, [620](#)
- vect\_size, [620](#)
- vect\_t, [612](#)
- vor, [619](#)
- vxor, [620](#)
- zero, [619](#)
- Simd128\_impl< true, true, false, 4 >::Converter, [451](#)
- t, [451](#)
- v, [451](#)
- Simd128\_impl< true, true, false, 8 >, [620](#)
- add, [627](#)
- addin, [628](#)
- aligned\_allocator, [622](#)
- aligned\_vector, [622](#)
- alignment, [631](#)
- blend, [627](#)
- compliant, [626](#)
- eq, [629](#)
- fmadd, [628](#)
- fmaddin, [628](#)
- fmaddx, [625](#)
- fmaddxin, [625](#)
- fmsub, [629](#)
- fmsubin, [629](#)
- fmsubx, [625](#)
- fmsubxin, [625](#)

- fnmadd, [628](#)
- fnmaddin, [628](#)
- fnmaddx, [625](#)
- fnmaddxin, [625](#)
- gather, [623](#)
- get, [626](#)
- greater, [624](#)
- greater\_eq, [624](#)
- hadd\_to\_scal, [625](#)
- is\_same\_element, [622](#)
- lesser, [624](#)
- lesser\_eq, [624](#)
- load, [623](#)
- loadu, [623](#)
- mask\_high, [629](#)
- mod, [629](#)
- mul, [628](#)
- mulhi, [624](#)
- mulhi\_fast, [629](#)
- mullo, [624](#)
- mulx, [625](#)
- pack, [627](#)
- pack\_even, [627](#)
- pack\_odd, [627](#)
- round, [629](#)
- scalar\_t, [622](#)
- set, [623](#)
- set1, [623](#)
- shuffle, [626](#)
- signbits, [630](#)
- sll, [626](#)
- sll128, [630](#)
- sra, [624](#)
- srl, [626](#)
- srl128, [630](#)
- store, [623](#)
- storeu, [623](#)
- stream, [624](#)
- sub, [628](#)
- subin, [628](#)
- transpose, [627](#)
- type\_string, [623](#)
- unpackhi, [627](#)
- unpackhi\_intrinsic, [626](#)
- unpacklo, [626](#)
- unpacklo\_intrinsic, [626](#)
- unpacklohi, [627](#)
- valid, [626](#)
- vand, [630](#)
- vandnot, [630](#)
- vect\_size, [630](#)
- vect\_t, [623](#)
- vor, [630](#)
- vxor, [630](#)
- zero, [630](#)
- Simd128\_impl< true, true, false, 8 >::Converter, [451](#)
  - t, [452](#)
  - v, [451](#)
- Simd128\_impl< true, true, true, 2 >, [631](#)
  - add, [636](#)
  - addin, [636](#)
  - aligned\_allocator, [633](#)
  - aligned\_vector, [633](#)
  - alignment, [641](#)
  - blend, [636](#)
  - compliant, [633](#)
  - eq, [639](#)
  - fmadd, [637](#)
  - fmaddin, [637](#)
  - fmaddx, [637](#)
  - fmaddxin, [637](#)
  - fmsub, [638](#)
  - fmsubin, [638](#)
  - fmsubx, [638](#)
  - fmsubxin, [639](#)
  - fnmadd, [638](#)
  - fnmaddin, [638](#)
  - fnmaddx, [638](#)
  - fnmaddxin, [638](#)
  - gather, [634](#)
  - greater, [639](#)
  - greater\_eq, [639](#)
  - hadd\_to\_scal, [639](#)
  - is\_same\_element, [633](#)
  - lesser, [639](#)
  - lesser\_eq, [639](#)
  - load, [634](#)
  - loadu, [634](#)
  - mod, [639](#)
  - mul, [637](#)
  - mulhi, [637](#)
  - mullo, [637](#)
  - mulx, [637](#)
  - pack, [636](#)
  - pack\_even, [635](#)
  - pack\_odd, [636](#)
  - round, [639](#)
  - scalar\_t, [633](#)
  - set, [633](#)
  - set1, [633](#)
  - shuffle, [635](#)
  - sll, [634](#)
  - sll128, [640](#)
  - sra, [635](#)
  - srl, [634](#)
  - srl128, [640](#)
  - store, [634](#)
  - storeu, [634](#)
  - stream, [634](#)
  - sub, [636](#)
  - subin, [636](#)
  - transpose, [636](#)
  - type\_string, [633](#)
  - unpackhi, [635](#)
  - unpackhi\_intrinsic, [635](#)
  - unpacklo, [635](#)

- unpacklo\_intrinsic, [635](#)
- unpacklohi, [635](#)
- valid, [633](#)
- vand, [640](#)
- vandnot, [640](#)
- vect\_size, [640](#)
- vect\_t, [633](#)
- vor, [640](#)
- vxor, [640](#)
- zero, [640](#)
- Simd128\_impl< true, true, true, 2 >::Converter, [452](#)
- t, [452](#)
- v, [452](#)
- Simd128\_impl< true, true, true, 4 >, [641](#)
- add, [646](#)
- addin, [646](#)
- aligned\_allocator, [643](#)
- aligned\_vector, [643](#)
- alignment, [650](#)
- blend, [646](#)
- compliant, [643](#)
- eq, [649](#)
- fmadd, [647](#)
- fmaddin, [647](#)
- fmaddx, [647](#)
- fmaddxin, [647](#)
- fmsub, [648](#)
- fmsubin, [648](#)
- fmsubx, [648](#)
- fmsubxin, [648](#)
- fnmadd, [647](#)
- fnmaddin, [648](#)
- fnmaddx, [648](#)
- fnmaddxin, [648](#)
- gather, [644](#)
- greater, [649](#)
- greater\_eq, [649](#)
- hadd\_to\_scal, [649](#)
- is\_same\_element, [643](#)
- lesser, [649](#)
- lesser\_eq, [649](#)
- load, [644](#)
- loadu, [644](#)
- mod, [649](#)
- mul, [647](#)
- mulhi, [647](#)
- mullo, [646](#)
- mulx, [647](#)
- pack, [646](#)
- pack\_even, [645](#)
- pack\_odd, [645](#)
- round, [649](#)
- scalar\_t, [643](#)
- set, [643](#)
- set1, [643](#)
- shuffle, [645](#)
- sll, [644](#)
- sll128, [650](#)
- sra, [644](#)
- srl, [644](#)
- srl128, [650](#)
- store, [644](#)
- storeu, [644](#)
- stream, [644](#)
- sub, [646](#)
- subin, [646](#)
- transpose, [646](#)
- type\_string, [643](#)
- unpackhi, [645](#)
- unpackhi\_intrinsic, [645](#)
- unpacklo, [645](#)
- unpacklo\_intrinsic, [645](#)
- unpacklohi, [645](#)
- valid, [643](#)
- vand, [650](#)
- vandnot, [650](#)
- vect\_size, [650](#)
- vect\_t, [643](#)
- vor, [650](#)
- vxor, [650](#)
- zero, [650](#)
- Simd128\_impl< true, true, true, 4 >::Converter, [452](#)
- t, [452](#)
- v, [452](#)
- Simd128\_impl< true, true, true, 8 >, [651](#)
- add, [656](#)
- addin, [656](#)
- aligned\_allocator, [653](#)
- aligned\_vector, [653](#)
- alignment, [661](#)
- blend, [656](#)
- compliant, [653](#)
- eq, [659](#)
- fmadd, [657](#)
- fmaddin, [657](#)
- fmaddx, [657](#)
- fmaddxin, [657](#)
- fmsub, [658](#)
- fmsubin, [658](#)
- fmsubx, [658](#)
- fmsubxin, [659](#)
- fnmadd, [658](#)
- fnmaddin, [658](#)
- fnmaddx, [658](#)
- fnmaddxin, [658](#)
- gather, [654](#)
- get, [654](#)
- greater, [659](#)
- greater\_eq, [659](#)
- hadd\_to\_scal, [659](#)
- is\_same\_element, [653](#)
- lesser, [659](#)
- lesser\_eq, [659](#)
- load, [654](#)
- loadu, [654](#)
- mask\_high, [659](#)

- mod, [660](#)
- mul, [657](#)
- mulhi, [657](#)
- mulhi\_fast, [660](#)
- mullo, [657](#)
- mulx, [657](#)
- pack, [656](#)
- pack\_even, [655](#)
- pack\_odd, [656](#)
- round, [659](#)
- scalar\_t, [653](#)
- set, [654](#)
- set1, [653](#)
- shuffle, [655](#)
- signbits, [660](#)
- sll, [654](#)
- sll128, [660](#)
- sra, [655](#)
- srl, [655](#)
- srl128, [660](#)
- store, [654](#)
- storeu, [654](#)
- stream, [654](#)
- sub, [656](#)
- subin, [656](#)
- transpose, [656](#)
- type\_string, [653](#)
- unpackhi, [655](#)
- unpackhi\_intrinsic, [655](#)
- unpacklo, [655](#)
- unpacklo\_intrinsic, [655](#)
- unpacklohi, [655](#)
- valid, [653](#)
- vand, [660](#)
- vandnot, [661](#)
- vect\_size, [661](#)
- vect\_t, [653](#)
- vor, [660](#)
- vxor, [661](#)
- zero, [660](#)
- Simd128\_impl< true, true, true, 8 >::Converter, [453](#)
- t, [453](#)
- v, [453](#)
- simd128\_int16.inl, [1118](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL, [1118](#)
- simd128\_int32.inl, [1118](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL, [1119](#)
- simd128\_int64.inl, [1119](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL, [1119](#)
- vect\_t, [1119](#)
- Simd128i\_base, [661](#)
- sll128, [662](#)
- srl128, [662](#)
- vand, [662](#)
- vandnot, [662](#)
- vect\_t, [662](#)
- vor, [662](#)
- vxor, [662](#)
- zero, [662](#)
- Simd256
- simd256.inl, [1120](#)
- simd256.inl, [1119](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL, [1120](#)
- Simd256, [1120](#)
- simd256\_double.inl, [1120](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL, [1120](#)
- simd256\_float.inl, [1121](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL, [1121](#)
- Simd256\_impl< ArithType, Int, Signed, Size >, [663](#)
- Simd256\_impl< true, false, true, 4 >, [663](#)
- Simd256\_impl< true, false, true, 8 >, [663](#)
- add, [668](#)
- addin, [668](#)
- aligned\_allocator, [665](#)
- aligned\_vector, [665](#)
- alignment, [671](#)
- blend, [668](#)
- blendv, [668](#)
- ceil, [671](#)
- compliant, [665](#)
- div, [669](#)
- eq, [669](#)
- floor, [671](#)
- fmadd, [669](#)
- fmaddin, [669](#)
- fmsub, [669](#)
- fmsubin, [669](#)
- fnmadd, [669](#)
- fnmaddin, [669](#)
- gather, [666](#)
- greater, [670](#)
- greater\_eq, [670](#)
- hadd, [671](#)
- hadd\_to\_scal, [671](#)
- is\_same\_element, [665](#)
- lesser, [670](#)
- lesser\_eq, [670](#)
- load, [666](#)
- loadu, [666](#)
- mod, [671](#)
- mul, [668](#)
- mulin, [668](#)
- pack, [667](#)
- pack\_even, [667](#)
- pack\_odd, [667](#)
- round, [671](#)
- scalar\_t, [665](#)
- set, [666](#)
- set1, [665](#)
- store, [666](#)

- storeu, [666](#)
- stream, [666](#)
- sub, [668](#)
- subin, [668](#)
- transpose, [667](#)
- type\_string, [665](#)
- unpackhi, [667](#)
- unpackhi\_intrinsic, [667](#)
- unpacklo, [667](#)
- unpacklo\_intrinsic, [666](#)
- unpacklohi, [667](#)
- valid, [665](#)
- vand, [670](#)
- vandnot, [670](#)
- vect\_size, [671](#)
- vect\_t, [665](#)
- vor, [670](#)
- vxor, [670](#)
- zero, [665](#)
- Simd256\_impl< true, false, true, 8 >::Converter, [453](#)
  - t, [453](#)
  - v, [453](#)
- Simd256\_impl< true, true, false, 2 >, [672](#)
  - add, [679](#)
  - addin, [679](#)
  - aligned\_allocator, [674](#)
  - aligned\_vector, [674](#)
  - alignment, [681](#)
  - blend, [679](#)
  - compliant, [677](#)
  - eq, [681](#)
  - fmadd, [680](#)
  - fmaddin, [680](#)
  - fmaddx, [676](#)
  - fmaddxin, [676](#)
  - fmsub, [680](#)
  - fmsubin, [681](#)
  - fmsubx, [677](#)
  - fmsubxin, [677](#)
  - fnmadd, [680](#)
  - fnmaddin, [680](#)
  - fnmaddx, [676](#)
  - fnmaddxin, [677](#)
  - gather, [675](#)
  - greater, [675](#)
  - greater\_eq, [676](#)
  - hadd\_to\_scal, [677](#)
  - half\_t, [674](#)
  - is\_same\_element, [674](#)
  - lesser, [676](#)
  - lesser\_eq, [676](#)
  - load, [675](#)
  - loadu, [675](#)
  - mod, [681](#)
  - mul, [680](#)
  - mulhi, [676](#)
  - mullo, [680](#)
  - mulx, [676](#)
  - pack, [678](#)
  - pack\_even, [678](#)
  - pack\_odd, [678](#)
  - round, [681](#)
  - scalar\_t, [674](#)
  - set, [674](#)
  - set1, [674](#)
  - shuffle, [678](#)
  - simdHalf, [674](#)
  - sll, [677](#)
  - sra, [675](#)
  - srl, [677](#)
  - store, [675](#)
  - storeu, [675](#)
  - stream, [675](#)
  - sub, [679](#)
  - subin, [680](#)
  - transpose, [679](#)
  - type\_string, [674](#)
  - unpackhi, [678](#)
  - unpackhi\_intrinsic, [678](#)
  - unpacklo, [678](#)
  - unpacklo\_intrinsic, [678](#)
  - unpacklohi, [678](#)
  - valid, [677](#)
  - vect\_size, [681](#)
  - vect\_t, [674](#)
  - zero, [681](#)
- Simd256\_impl< true, true, false, 2 >::Converter, [453](#)
  - t, [454](#)
  - v, [453](#)
- Simd256\_impl< true, true, false, 4 >, [682](#)
  - add, [696](#)
  - addin, [696](#)
  - aligned\_allocator, [685](#)
  - aligned\_vector, [685](#)
  - alignment, [701](#)
  - blend, [696](#)
  - compliant, [692](#)
  - eq, [699](#)
  - fmadd, [697](#), [698](#)
  - fmaddin, [698](#)
  - fmaddx, [688](#), [691](#)
  - fmaddxin, [688](#), [691](#)
  - fmsub, [699](#)
  - fmsubin, [699](#)
  - fmsubx, [688](#), [691](#)
  - fmsubxin, [689](#), [691](#)
  - fnmadd, [698](#)
  - fnmaddin, [698](#)
  - fnmaddx, [688](#), [691](#)
  - fnmaddxin, [688](#), [691](#)
  - gather, [686](#), [689](#)
  - greater, [687](#), [690](#)
  - greater\_eq, [687](#), [690](#)
  - hadd\_to\_scal, [689](#), [692](#)
  - half\_t, [686](#)
  - is\_same\_element, [685](#)

- lesser, [687](#), [690](#)
- lesser\_eq, [688](#), [690](#)
- load, [687](#), [689](#)
- loadu, [687](#), [689](#)
- mod, [700](#)
- mul, [697](#)
- mulhi, [688](#), [690](#)
- mullo, [697](#)
- mulx, [688](#), [691](#)
- pack, [695](#)
- pack\_even, [694](#), [695](#)
- pack\_odd, [695](#)
- round, [699](#)
- scalar\_t, [685](#)
- set, [686](#), [689](#), [692](#)
- set1, [686](#), [689](#)
- shuffle, [693](#)
- shuffle\_twice, [693](#)
- simdHalf, [685](#), [686](#)
- sll, [692](#)
- sra, [687](#), [690](#)
- srl, [693](#)
- store, [687](#), [690](#)
- storeu, [687](#), [690](#)
- stream, [687](#), [690](#)
- sub, [697](#)
- subin, [697](#)
- transpose, [695](#)
- type\_string, [686](#), [689](#)
- unpackhi, [694](#)
- unpackhi\_intrinsic, [693](#), [694](#)
- unpacklo, [694](#)
- unpacklo\_intrinsic, [693](#)
- unpacklohi, [694](#)
- valid, [692](#)
- vand, [700](#)
- vandnot, [701](#)
- vect\_size, [701](#)
- vect\_t, [686](#)
- vor, [700](#)
- vxor, [700](#)
- zero, [700](#)
- Simd256\_impl< true, true, false, 4 >::Converter, [454](#)
  - t, [454](#)
  - v, [454](#)
- Simd256\_impl< true, true, false, 8 >, [701](#)
  - add, [708](#)
  - addin, [709](#)
  - aligned\_allocator, [703](#)
  - aligned\_vector, [703](#)
  - alignment, [711](#)
  - blend, [708](#)
  - compliant, [707](#)
  - eq, [710](#)
  - fmadd, [709](#)
  - fmaddin, [709](#)
  - fmaddx, [706](#)
  - fmaddxin, [706](#)
  - fmsub, [710](#)
  - fmsubin, [710](#)
  - fmsubx, [706](#)
  - fmsubxin, [706](#)
  - fnmadd, [709](#)
  - fnmaddin, [709](#)
  - fnmaddx, [706](#)
  - fnmaddxin, [706](#)
  - gather, [704](#)
  - get, [707](#)
  - greater, [705](#)
  - greater\_eq, [705](#)
  - hadd\_to\_scal, [706](#)
  - half\_t, [703](#)
  - is\_same\_element, [703](#)
  - lesser, [705](#)
  - lesser\_eq, [705](#)
  - load, [704](#)
  - loadu, [704](#)
  - mask\_high, [710](#)
  - mod, [710](#)
  - mul, [709](#)
  - mulhi, [705](#)
  - mulhi\_fast, [710](#)
  - mullo, [705](#)
  - mulx, [705](#)
  - pack, [708](#)
  - pack\_even, [708](#)
  - pack\_odd, [708](#)
  - round, [710](#)
  - scalar\_t, [703](#)
  - set, [704](#)
  - set1, [704](#)
  - shuffle, [707](#)
  - signbits, [710](#)
  - simdHalf, [703](#)
  - sll, [707](#)
  - sra, [705](#)
  - srl, [707](#)
  - store, [704](#)
  - storeu, [704](#)
  - stream, [704](#)
  - sub, [709](#)
  - subin, [709](#)
  - transpose, [708](#)
  - type\_string, [704](#)
  - unpackhi, [707](#)
  - unpackhi\_intrinsic, [707](#)
  - unpacklo, [707](#)
  - unpacklo\_intrinsic, [707](#)
  - unpacklohi, [708](#)
  - valid, [706](#)
  - vect\_size, [711](#)
  - vect\_t, [703](#)
  - zero, [711](#)
- Simd256\_impl< true, true, false, 8 >::Converter, [454](#)
  - t, [454](#)
  - v, [454](#)

Simd256\_impl< true, true, true, 2 >, 711  
   add, 717  
   addin, 717  
   aligned\_allocator, 713  
   aligned\_vector, 713  
   alignment, 721  
   blend, 717  
   compliant, 714  
   eq, 720  
   fmadd, 718  
   fmaddin, 718  
   fmaddx, 718  
   fmaddxin, 718  
   fmsub, 719  
   fmsubin, 719  
   fmsubx, 719  
   fmsubxin, 719  
   fnmadd, 718  
   fnmaddin, 719  
   fnmaddx, 719  
   fnmaddxin, 719  
   gather, 714  
   greater, 720  
   greater\_eq, 720  
   hadd\_to\_scal, 720  
   half\_t, 713  
   is\_same\_element, 713  
   lesser, 720  
   lesser\_eq, 720  
   load, 714  
   loadu, 714  
   mod, 720  
   mul, 718  
   mulhi, 718  
   mullo, 717  
   mulx, 718  
   pack, 716  
   pack\_even, 716  
   pack\_odd, 716  
   round, 720  
   scalar\_t, 713  
   set, 714  
   set1, 714  
   shuffle, 715  
   simdHalf, 713  
   sll, 715  
   sra, 715  
   srl, 715  
   store, 715  
   storeu, 715  
   stream, 715  
   sub, 717  
   subin, 717  
   transpose, 716  
   type\_string, 714  
   unpackhi, 716  
   unpackhi\_intrinsic, 715  
   unpacklo, 716

  unpacklo\_intrinsic, 715  
   unpacklohi, 716  
   valid, 714  
   vect\_size, 721  
   vect\_t, 713  
   zero, 721  
 Simd256\_impl< true, true, true, 2 >::Converter, 455  
   t, 455  
   v, 455  
 Simd256\_impl< true, true, true, 4 >, 721  
   add, 729, 736  
   addin, 729, 736  
   aligned\_allocator, 725  
   aligned\_vector, 725  
   alignment, 740  
   blend, 729, 736  
   compliant, 726, 732  
   eq, 731, 738  
   fmadd, 730, 737  
   fmaddin, 730, 737  
   fmaddx, 730, 737  
   fmaddxin, 730, 737  
   fmsub, 731, 738  
   fmsubin, 731, 738  
   fmsubx, 731, 738  
   fmsubxin, 731, 738  
   fnmadd, 730, 737  
   fnmaddin, 730, 737  
   fnmaddx, 730, 738  
   fnmaddxin, 731, 738  
   gather, 726, 733  
   greater, 731, 738  
   greater\_eq, 732, 739  
   hadd\_to\_scal, 732, 739  
   half\_t, 724, 725  
   is\_same\_element, 725  
   lesser, 732, 739  
   lesser\_eq, 732, 739  
   load, 726, 733  
   loadu, 726, 733  
   mod, 732, 739  
   mul, 729, 736  
   mulhi, 729, 736  
   mullo, 729, 736  
   mulx, 730, 737  
   pack, 728, 735  
   pack\_even, 728, 735  
   pack\_odd, 728, 735  
   round, 732, 739  
   scalar\_t, 724, 725  
   set, 726, 733  
   set1, 726, 733  
   shuffle, 727, 734  
   shuffle\_twice, 727, 734  
   simdHalf, 725  
   sll, 727, 734  
   sra, 727, 734  
   srl, 727, 734

- store, [726](#), [733](#)
- storeu, [727](#), [733](#)
- stream, [727](#), [734](#)
- sub, [729](#), [736](#)
- subin, [729](#), [736](#)
- transpose, [728](#), [735](#)
- type\_string, [726](#), [732](#)
- unpackhi, [728](#), [735](#)
- unpackhi\_intrinsic, [727](#), [734](#)
- unpacklo, [728](#), [734](#)
- unpacklo\_intrinsic, [727](#), [734](#)
- unpacklohi, [728](#), [735](#)
- valid, [726](#), [732](#)
- vand, [740](#)
- vandnot, [740](#)
- vect\_size, [740](#)
- vect\_t, [724](#), [725](#)
- vor, [740](#)
- vxor, [740](#)
- zero, [739](#)
- Simd256\_impl< true, true, true, 4 >::Converter, [455](#)
- t, [455](#)
- v, [455](#)
- Simd256\_impl< true, true, true, 8 >, [740](#)
- add, [746](#)
- addin, [746](#)
- aligned\_allocator, [743](#)
- aligned\_vector, [743](#)
- alignment, [750](#)
- blend, [746](#)
- compliant, [743](#)
- eq, [749](#)
- fmadd, [747](#)
- fmaddin, [747](#)
- fmaddx, [747](#)
- fmaddxin, [747](#)
- fmsub, [748](#)
- fmsubin, [748](#)
- fmsubx, [748](#)
- fmsubxin, [748](#)
- fnmadd, [748](#)
- fnmaddin, [748](#)
- fnmaddx, [748](#)
- fnmaddxin, [748](#)
- gather, [744](#)
- get, [744](#)
- greater, [749](#)
- greater\_eq, [749](#)
- hadd\_to\_scal, [749](#)
- half\_t, [742](#)
- is\_same\_element, [743](#)
- lesser, [749](#)
- lesser\_eq, [749](#)
- load, [744](#)
- loadu, [744](#)
- mask\_high, [749](#)
- mod, [750](#)
- mul, [747](#)
- mulhi, [747](#)
- mulhi\_fast, [749](#)
- mullo, [747](#)
- mulx, [747](#)
- pack, [746](#)
- pack\_even, [745](#)
- pack\_odd, [746](#)
- round, [749](#)
- scalar\_t, [743](#)
- set, [743](#)
- set1, [743](#)
- shuffle, [745](#)
- signbits, [750](#)
- simdHalf, [743](#)
- sll, [744](#)
- sra, [745](#)
- srl, [744](#)
- store, [744](#)
- storeu, [744](#)
- stream, [744](#)
- sub, [746](#)
- subin, [746](#)
- transpose, [746](#)
- type\_string, [743](#)
- unpackhi, [745](#)
- unpackhi\_intrinsic, [745](#)
- unpacklo, [745](#)
- unpacklo\_intrinsic, [745](#)
- unpacklohi, [745](#)
- valid, [743](#)
- vect\_size, [750](#)
- vect\_t, [742](#)
- zero, [750](#)
- Simd256\_impl< true, true, true, 8 >::Converter, [455](#)
- t, [456](#)
- v, [455](#)
- simd256\_int16.inl, [1121](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL, [1121](#)
- simd256\_int32.inl, [1121](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL, [1122](#)
- simd256\_int64.inl, [1122](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL, [1122](#)
- vect\_t, [1122](#)
- Simd256fp\_base, [750](#)
- Simd256i\_base, [751](#)
- vect\_t, [751](#)
- zero, [751](#)
- Simd512
- simd512.inl, [1123](#)
- simd512.inl, [1123](#)
- \_\_FFLASFFPACK\_simd512\_INL, [1123](#)
- Simd512, [1123](#)
- simd512\_double.inl, [1123](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL, [1123](#)
- simd512\_float.inl, [1124](#)

- [\\_\\_FFLASFFPACK\\_simd512\\_float\\_INL](#), 1124
- [Simd512\\_impl< ArithType, Int, Signed, Size >](#), 751
- [Simd512\\_impl< true, false, true, 4 >](#), 751
- [Simd512\\_impl< true, false, true, 8 >](#), 751
  - [add](#), 756
  - [addin](#), 756
  - [aligned\\_allocator](#), 753
  - [aligned\\_vector](#), 753
  - [alignment](#), 759
  - [blend](#), 756
  - [blendv](#), 756
  - [ceil](#), 759
  - [compliant](#), 753
  - [div](#), 757
  - [eq](#), 758
  - [floor](#), 758
  - [fmadd](#), 757
  - [fmaddin](#), 757
  - [fmsub](#), 758
  - [fmsubin](#), 758
  - [fnmadd](#), 757
  - [fnmaddin](#), 757
  - [gather](#), 754
  - [greater](#), 758
  - [greater\\_eq](#), 758
  - [hadd](#), 759
  - [hadd\\_to\\_scal](#), 759
  - [is\\_same\\_element](#), 753
  - [lesser](#), 758
  - [lesser\\_eq](#), 758
  - [load](#), 754
  - [loadu](#), 754
  - [mul](#), 757
  - [mulin](#), 757
  - [pack](#), 756
  - [pack\\_even](#), 755
  - [pack\\_odd](#), 755
  - [round](#), 759
  - [scalar\\_t](#), 753
  - [set](#), 754
  - [set1](#), 754
  - [shuffle](#), 755
  - [store](#), 754
  - [storeu](#), 754
  - [stream](#), 754
  - [sub](#), 756
  - [subin](#), 757
  - [transpose](#), 756
  - [type\\_string](#), 753
  - [unpackhi](#), 755
  - [unpackhi\\_intrinsic](#), 755
  - [unpacklo](#), 755
  - [unpacklo\\_intrinsic](#), 755
  - [unpacklohi](#), 755
  - [valid](#), 753
  - [vect\\_size](#), 759
  - [vect\\_t](#), 753
  - [zero](#), 753
- [Simd512\\_impl< true, true, false, 8 >](#), 759
  - [add](#), 767
  - [addin](#), 767
  - [aligned\\_allocator](#), 761
  - [aligned\\_vector](#), 762
  - [alignment](#), 770
  - [blend](#), 767
  - [compliant](#), 765
  - [eq](#), 769
  - [fmadd](#), 768
  - [fmaddin](#), 768
  - [fmaddx](#), 764
  - [fmaddxin](#), 764
  - [fmsub](#), 768
  - [fmsubin](#), 768
  - [fmsubx](#), 765
  - [fmsubxin](#), 765
  - [fnmadd](#), 768
  - [fnmaddin](#), 768
  - [fnmaddx](#), 764
  - [fnmaddxin](#), 765
  - [gather](#), 762
  - [greater](#), 763
  - [greater\\_eq](#), 764
  - [hadd\\_to\\_scal](#), 765
  - [half\\_t](#), 762
  - [is\\_same\\_element](#), 762
  - [lesser](#), 763
  - [lesser\\_eq](#), 764
  - [load](#), 763
  - [loadu](#), 763
  - [mask\\_high](#), 769
  - [maskstore](#), 763
  - [mod](#), 769
  - [mul](#), 768
  - [mulhi](#), 764
  - [mulhi\\_fast](#), 769
  - [mullo](#), 764
  - [mulx](#), 764
  - [pack](#), 767
  - [pack\\_even](#), 766
  - [pack\\_odd](#), 767
  - [round](#), 769
  - [scalar\\_t](#), 761
  - [set](#), 762, 765
  - [set1](#), 762
  - [shuffle](#), 766
  - [signbits](#), 769
  - [simdHalf](#), 762
  - [sll](#), 765
  - [sra](#), 763
  - [srl](#), 766
  - [store](#), 763
  - [storeu](#), 763
  - [stream](#), 763
  - [sub](#), 767
  - [subin](#), 767
  - [transpose](#), 767

- type\_string, 762
- unpackhi, 766
- unpackhi\_intrinsic, 766
- unpacklo, 766
- unpacklo\_intrinsic, 766
- unpacklohi, 766
- valid, 765
- vand, 770
- vandnot, 770
- vect\_size, 770
- vect\_t, 762
- vor, 769
- vxor, 769
- zero, 769
- Simd512\_impl< true, true, false, 8 >::Converter, 456
  - t, 456
  - v, 456
- Simd512\_impl< true, true, true, 8 >, 770
  - add, 776
  - addin, 776
  - aligned\_allocator, 772
  - aligned\_vector, 773
  - alignment, 781
  - blend, 776
  - compliant, 773
  - eq, 779
  - fmadd, 777
  - fmaddin, 777
  - fmaddx, 777
  - fmaddxin, 777
  - fmsub, 778
  - fmsubin, 778
  - fmsubx, 778
  - fmsubxin, 779
  - fnmadd, 778
  - fnmaddin, 778
  - fnmaddx, 778
  - fnmaddxin, 778
  - gather, 774
  - greater, 779
  - greater\_eq, 779
  - hadd\_to\_scal, 779
  - half\_t, 772
  - is\_same\_element, 773
  - lesser, 779
  - lesser\_eq, 779
  - load, 774
  - loadu, 774
  - mask\_high, 779
  - maskstore, 774
  - mod, 780
  - mul, 777
  - mulhi, 777
  - mulhi\_fast, 780
  - mullo, 777
  - mulx, 777
  - pack, 776
  - pack\_even, 775
  - pack\_odd, 776
  - round, 779
  - scalar\_t, 772
  - set, 773
  - set1, 773
  - shuffle, 775
  - signbits, 780
  - simdHalf, 772
  - sll, 774
  - sra, 775
  - srl, 774
  - store, 774
  - storeu, 774
  - stream, 774
  - sub, 776
  - subin, 776
  - transpose, 776
  - type\_string, 773
  - unpackhi, 775
  - unpackhi\_intrinsic, 775
  - unpacklo, 775
  - unpacklo\_intrinsic, 775
  - unpacklohi, 775
  - valid, 773
  - vand, 780
  - vandnot, 780
  - vect\_size, 781
  - vect\_t, 772
  - vor, 780
  - vxor, 780
  - zero, 780
- Simd512\_impl< true, true, true, 8 >::Converter, 456
  - t, 456
  - v, 456
- simd512\_int32.inl, 1124
  - \_\_FFLASFFPACK\_simd512\_int32\_INL, 1124
- simd512\_int64.inl, 1125
  - \_simd512\_int64\_INL, 1125
  - vect\_t, 1125
- Simd512i\_base, 781
  - vand, 782
  - vandnot, 782
  - vect\_t, 781
  - vor, 782
  - vxor, 782
  - zero, 781
- SIMD\_INT
  - fflas\_simd.h, 985
- simd\_modular.inl, 1125
- SimdChooser< T, bool, bool >, 782
- SimdChooser< T, false, b >, 782
  - value, 782
- SimdChooser< T, true, false >, 783
  - value, 783
- SimdChooser< T, true, true >, 783
  - value, 783
- simdHalf
  - Simd256\_impl< true, true, false, 2 >, 674

- Simd256\_impl< true, true, false, 4 >, [685](#), [686](#)
- Simd256\_impl< true, true, false, 8 >, [703](#)
- Simd256\_impl< true, true, true, 2 >, [713](#)
- Simd256\_impl< true, true, true, 4 >, [725](#)
- Simd256\_impl< true, true, true, 8 >, [743](#)
- Simd512\_impl< true, true, false, 8 >, [762](#)
- Simd512\_impl< true, true, true, 8 >, [772](#)
- SimdSparseMatrix
  - FFLAS, [76](#)
- simdToType< T >, [783](#)
- Single, [783](#)
- size
  - BlockTransposeSIMD< Field, Simd, >, [436](#)
  - Info, [508](#), [509](#)
  - RNSInteger< RNS >, [579](#)
  - RNSIntegerMod< RNS >, [584](#)
- SIZEOF\_\_\_INT64\_T
  - config.h, [877](#)
- SIZEOF\_CHAR
  - config.h, [876](#)
- SIZEOF\_INT
  - config.h, [877](#)
- SIZEOF\_LONG
  - config.h, [877](#)
- SIZEOF\_LONG\_LONG
  - config.h, [877](#)
- SIZEOF\_SHORT
  - config.h, [877](#)
- sll
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [598](#)
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, true, false, 2 >, [677](#)
  - Simd256\_impl< true, true, false, 4 >, [692](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, true, false, 8 >, [765](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- sll128
  - Simd128\_impl< true, true, false, 2 >, [609](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd128i\_base, [662](#)
- Solve
  - FFPACK, [348](#), [394](#)
- solve.C, [1125](#)
  - main, [1125](#)
- Solve\_modular\_double
  - ffpack.C, [1011](#)
  - ffpack\_c.h, [1044](#)
- solveLB
  - FFPACK, [368](#), [395](#)
- solveLB2
  - FFPACK, [369](#), [395](#)
- solveLB2\_modular\_double
  - ffpack.C, [1011](#)
  - ffpack\_c.h, [1044](#)
- solveLB\_modular\_double
  - ffpack.C, [1011](#)
  - ffpack\_c.h, [1044](#)
- Sparse< \_Field, SparseMatrix\_t::COO >, [784](#)
  - col, [784](#)
  - dat, [784](#)
  - delayed, [784](#)
  - Field, [784](#)
  - kmax, [785](#)
  - m, [785](#)
  - maxrow, [785](#)
  - n, [785](#)
  - nElements, [785](#)
  - nnz, [785](#)
  - row, [784](#)
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [785](#)
  - col, [786](#)
  - cst, [786](#)
  - dat, [786](#)
  - delayed, [786](#)
  - Field, [786](#)
  - kmax, [786](#)
  - m, [786](#)
  - maxrow, [787](#)
  - n, [786](#)
  - nElements, [787](#)
  - nnz, [786](#)
  - row, [786](#)
- Sparse< \_Field, SparseMatrix\_t::CSR >, [787](#)
  - col, [788](#)
  - dat, [788](#)
  - delayed, [788](#)
  - Field, [787](#)
  - kmax, [788](#)
  - m, [788](#)
  - maxrow, [788](#)
  - n, [788](#)
  - nElements, [788](#)
  - nnz, [788](#)
  - st, [788](#)
  - stend, [788](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - col, [789](#)
  - dat, [789](#)
  - delayed, [789](#)
  - Field, [789](#)
  - kmax, [789](#)
  - m, [790](#)
  - maxrow, [790](#)

- n, [790](#)
- nElements, [790](#)
- nMOnes, [790](#)
- nnz, [790](#)
- nOnes, [790](#)
- nOthers, [790](#)
- st, [789](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [790](#)
  - col, [792](#)
  - cst, [791](#)
  - dat, [792](#)
  - delayed, [791](#)
  - Field, [791](#)
  - kmax, [791](#)
  - m, [791](#)
  - maxrow, [792](#)
  - n, [791](#)
  - nElements, [791](#)
  - nnz, [791](#)
  - st, [792](#)
  - stend, [792](#)
- Sparse< \_Field, SparseMatrix\_t::ELL >, [792](#)
  - col, [793](#)
  - dat, [793](#)
  - delayed, [793](#)
  - Field, [793](#)
  - kmax, [793](#)
  - ld, [793](#)
  - m, [793](#)
  - maxrow, [793](#)
  - n, [793](#)
  - nElements, [793](#)
  - nnz, [793](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [794](#)
  - chunk, [794](#)
  - col, [795](#)
  - dat, [795](#)
  - delayed, [794](#)
  - kmax, [795](#)
  - ld, [794](#)
  - m, [794](#)
  - maxrow, [795](#)
  - n, [794](#)
  - nChunks, [795](#)
  - nElements, [795](#)
  - nnz, [795](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [795](#)
  - chunk, [796](#)
  - col, [797](#)
  - cst, [796](#)
  - dat, [797](#)
  - delayed, [796](#)
  - kmax, [796](#)
  - ld, [796](#)
  - m, [796](#)
  - maxrow, [797](#)
  - n, [796](#)
  - nChunks, [797](#)
- nElements, [796](#)
- nnz, [796](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [797](#)
  - col, [798](#)
  - cst, [798](#)
  - dat, [798](#)
  - delayed, [798](#)
  - Field, [798](#)
  - kmax, [798](#)
  - ld, [798](#)
  - m, [798](#)
  - maxrow, [798](#)
  - n, [798](#)
  - nElements, [798](#)
  - nnz, [798](#)
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [799](#)
  - dat, [800](#)
  - delayed, [799](#)
  - Field, [799](#)
  - kmax, [799](#)
  - m, [799](#)
  - maxrow, [800](#)
  - mone, [800](#)
  - n, [800](#)
  - nElements, [800](#)
  - nnz, [800](#)
  - one, [800](#)
  - Self\_t, [799](#)
- Sparse< \_Field, SparseMatrix\_t::SELL >, [800](#)
  - chunk, [801](#)
  - chunkSize, [802](#)
  - col, [802](#)
  - dat, [802](#)
  - delayed, [801](#)
  - Field, [801](#)
  - kmax, [801](#)
  - m, [801](#)
  - maxrow, [801](#)
  - n, [801](#)
  - nChunks, [802](#)
  - nElements, [802](#)
  - nnz, [802](#)
  - perm, [802](#)
  - sigma, [801](#)
  - st, [802](#)
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [802](#)
  - chunk, [803](#)
  - chunkSize, [804](#)
  - col, [804](#)
  - cst, [803](#)
  - dat, [804](#)
  - delayed, [803](#)
  - Field, [803](#)
  - kmax, [803](#)
  - m, [803](#)
  - maxrow, [804](#)
  - n, [803](#)
  - nChunks, [804](#)

- nElements, [804](#)
  - nnz, [804](#)
  - perm, [804](#)
  - sigma, [804](#)
  - st, [804](#)
- Sparse< Field, SparseMatrix\_t, IdxT, PtrT >, [783](#)
- sparse\_delete
  - FFLAS, [154](#), [155](#), [157–159](#), [162](#)
- sparse\_init
  - FFLAS, [155–160](#), [162](#), [163](#)
- sparse\_matrix\_traits.h, [1126](#)
- sparse\_print
  - FFLAS, [156](#), [159](#), [162](#)
- SparseMatrix\_t
  - FFLAS, [81](#)
- SpecRankProfile
  - FFPACK, [393](#)
- SpecRankProfile\_modular\_double
  - ffpack.C, [1010](#)
  - ffpack\_c.h, [1043](#)
- splitt
  - parallel.h, [1100](#)
- SPLITTER
  - parallel.h, [1101](#)
- splitting\_0
  - parallel.h, [1100](#)
- splitting\_1
  - parallel.h, [1100](#)
- splitting\_2
  - parallel.h, [1100](#)
- splitting\_3
  - parallel.h, [1100](#)
- SpMat< Field, flag >, [804](#)
  - \_coo, [805](#)
  - \_csr, [805](#)
  - \_ell, [805](#)
- square\_inplace
  - FFLAS::\_ftranspose\_impl, [203](#)
- sra
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [598](#)
  - Simd128\_impl< true, true, false, 2 >, [604](#)
  - Simd128\_impl< true, true, false, 4 >, [614](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [705](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- srl
  - ScalFunctionsBase< Element, typename enable\_if< is\_integral< Element >::value >::type >, [598](#)
- Simd128\_impl< true, true, false, 2 >, [606](#)
- Simd128\_impl< true, true, false, 4 >, [616](#)
- Simd128\_impl< true, true, false, 8 >, [626](#)
- Simd128\_impl< true, true, true, 2 >, [634](#)
- Simd128\_impl< true, true, true, 4 >, [644](#)
- Simd128\_impl< true, true, true, 8 >, [655](#)
- Simd256\_impl< true, true, false, 2 >, [677](#)
- Simd256\_impl< true, true, false, 4 >, [693](#)
- Simd256\_impl< true, true, false, 8 >, [707](#)
- Simd256\_impl< true, true, true, 2 >, [715](#)
- Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
- Simd256\_impl< true, true, true, 8 >, [744](#)
- Simd512\_impl< true, true, false, 8 >, [766](#)
- Simd512\_impl< true, true, true, 8 >, [774](#)
- srl128
  - Simd128\_impl< true, true, false, 2 >, [609](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd128i\_base, [662](#)
- sscal\_
  - config-blas.h, [871](#)
- ST
  - fflas\_transpose.h, [994](#)
- st
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [789](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [792](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [802](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [804](#)
- StatsMatrix, [805](#)
  - averageCol, [807](#)
  - averageColDifference, [807](#)
  - averageRow, [806](#)
  - averageRowDifference, [807](#)
  - coldim, [806](#)
  - denseCols, [808](#)
  - denseRows, [808](#)
  - deviationCol, [807](#)
  - deviationColDifference, [807](#)
  - deviationRow, [806](#)
  - deviationRowDifference, [807](#)
  - maxCol, [806](#)
  - maxColDifference, [807](#)
  - maxRow, [806](#)
  - maxRowDifference, [807](#)
  - minCol, [807](#)
  - minColDifference, [807](#)
  - minRow, [806](#)
  - minRowDifference, [807](#)
  - nDenseCols, [808](#)
  - nDenseRows, [807](#)
  - nEmptyCols, [808](#)
  - nEmptyColsEnd, [808](#)
  - nEmptyRows, [808](#)

- nMOnes, [806](#)
- nnz, [806](#)
- nOnes, [806](#)
- nOthers, [806](#)
- rowdim, [806](#)
- STDC\_HEADERS
  - config.h, [877](#)
- stend
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [788](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [792](#)
- store
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [726](#), [733](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, false, true, 8 >, [754](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- storeu
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [733](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
  - Simd512\_impl< true, false, true, 8 >, [754](#)
  - Simd512\_impl< true, true, false, 8 >, [763](#)
  - Simd512\_impl< true, true, true, 8 >, [774](#)
- stream
  - Simd128\_impl< true, true, false, 2 >, [603](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [624](#)
  - Simd128\_impl< true, true, true, 2 >, [634](#)
  - Simd128\_impl< true, true, true, 4 >, [644](#)
  - Simd128\_impl< true, true, true, 8 >, [654](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [675](#)
  - Simd256\_impl< true, true, false, 4 >, [687](#), [690](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [744](#)
- Simd512\_impl< true, false, true, 8 >, [754](#)
- Simd512\_impl< true, true, false, 8 >, [763](#)
- Simd512\_impl< true, true, true, 8 >, [774](#)
- strmm\_
  - config-blas.h, [872](#)
- strsm\_
  - config-blas.h, [871](#)
- sub
  - FFLAS::vectorised, [299](#)
  - FieldSimd< \_Field >, [477](#)
  - RNSIntegerMod< RNS >, [585](#)
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [607](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd256\_impl< true, true, false, 2 >, [679](#)
  - Simd256\_impl< true, true, false, 4 >, [697](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [717](#)
  - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [756](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- sub\_r
  - FieldSimd< \_Field >, [478](#)
- subin
  - FieldSimd< \_Field >, [478](#)
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [608](#)
  - Simd128\_impl< true, true, false, 4 >, [617](#)
  - Simd128\_impl< true, true, false, 8 >, [628](#)
  - Simd128\_impl< true, true, true, 2 >, [636](#)
  - Simd128\_impl< true, true, true, 4 >, [646](#)
  - Simd128\_impl< true, true, true, 8 >, [656](#)
  - Simd256\_impl< true, false, true, 8 >, [668](#)
  - Simd256\_impl< true, true, false, 2 >, [680](#)
  - Simd256\_impl< true, true, false, 4 >, [697](#)
  - Simd256\_impl< true, true, false, 8 >, [709](#)
  - Simd256\_impl< true, true, true, 2 >, [717](#)
  - Simd256\_impl< true, true, true, 4 >, [729](#), [736](#)
  - Simd256\_impl< true, true, true, 8 >, [746](#)
  - Simd512\_impl< true, false, true, 8 >, [757](#)
  - Simd512\_impl< true, true, false, 8 >, [767](#)
  - Simd512\_impl< true, true, true, 8 >, [776](#)
- subin\_r
  - FieldSimd< \_Field >, [478](#)
- subp
  - FFLAS::vectorised, [299](#)
- support\_fast\_mod< double >, [808](#)
- support\_fast\_mod< float >, [809](#)
- support\_fast\_mod< int64\_t >, [809](#)
- support\_fast\_mod< T >, [808](#)
- support\_simd< T >, [809](#)

- support\_simd\_add< T >, [810](#)
- support\_simd\_mod< T >, [810](#)
- swapval
  - FFPACK, [407](#)
- SYNCH\_GROUP
  - parallel.h, [1095](#)
- SysTimer
  - FFLAS, [78](#)
- T
  - limits< char >, [518](#)
  - limits< double >, [519](#)
  - limits< float >, [519](#)
  - limits< Givaro::Integer >, [520](#)
  - limits< int >, [521](#)
  - limits< long >, [522](#)
  - limits< long long >, [522](#)
  - limits< Reclnt::rint< K > >, [523](#)
  - limits< Reclnt::ruint< K > >, [524](#)
  - limits< short int >, [524](#)
  - limits< signed char >, [525](#)
  - limits< unsigned char >, [526](#)
  - limits< unsigned int >, [526](#)
  - limits< unsigned long >, [527](#)
  - limits< unsigned long long >, [528](#)
  - limits< unsigned short int >, [529](#)
- t
  - Simd128\_impl< true, true, false, 2 >::Converter, [451](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [451](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [452](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [452](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [452](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [453](#)
  - Simd256\_impl< true, false, true, 8 >::Converter, [453](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [454](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [454](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [454](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [455](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [455](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [456](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [456](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [456](#)
- TASK
  - parallel.h, [1095](#)
- tBC
  - test-lu.C, [1167](#)
  - test-permutations.C, [1172](#)
- Test
  - Test< Elt >, [812](#)
- test
  - test-maxdelayeddim.C, [1168](#)
- Test< Elt >, [810](#)
  - \_mm, [813](#)
  - \_nn, [813](#)
  - cardinality, [812](#)
  - doTests, [812](#)
  - Elt\_ptr, [811](#)
  - enable\_if\_no\_simd\_t, [811](#)
  - enable\_if\_simd128\_t, [811](#)
  - enable\_if\_simd256\_t, [811](#)
  - enable\_if\_simd512\_t, [812](#)
  - enable\_if\_t, [811](#)
  - F, [812](#)
  - Field, [811](#)
  - is\_same\_element, [811](#)
  - Residu, [811](#)
  - run, [812](#)
  - Test, [812](#)
  - test\_ftranspose, [812](#)
- test-charpoly-check.C, [1127](#)
  - ENABLE\_CHECKER\_charpoly, [1127](#)
  - main, [1128](#)
  - printPolynomial, [1128](#)
  - TIME\_CHECKER\_CHARPOLY, [1128](#)
- test-charpoly.C, [1128](#)
  - launch\_test, [1128](#)
  - main, [1129](#)
  - run\_with\_field, [1129](#)
- test-compressQ.C, [1129](#)
  - Field, [1129](#)
  - main, [1130](#)
  - printvect, [1130](#)
- test-det-check.C, [1130](#)
  - ENABLE\_CHECKER\_Det, [1130](#)
  - main, [1130](#)
  - TIME\_CHECKER\_Det, [1130](#)
- test-det.C, [1131](#)
  - main, [1131](#)
  - test\_det, [1131](#)
- test-echelon.C, [1131](#)
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, [1132](#)
  - \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, [1132](#)
  - \_\_FFLASFFPACK\_SEQUENTIAL, [1132](#)
  - main, [1134](#)
  - run\_with\_field, [1133](#)
  - test\_colechelon, [1132](#)
  - test\_redcoechelon, [1133](#)
  - test\_redrowechelon, [1133](#)
  - test\_rowechelon, [1133](#)
- test-fadd.C, [1134](#)
  - main, [1135](#)

- test\_fadd, 1134
- test\_faddin, 1134
- test\_fsub, 1135
- test\_fsubin, 1135
- test-fdot.C, 1135
  - check\_fdot, 1136
  - ENABLE\_ALL\_CHECKINGS, 1136
  - main, 1136
  - run\_with\_field, 1136
  - run\_with\_Integer, 1136
- test-fgemm-check.C, 1136
  - ENABLE\_ALL\_CHECKINGS, 1137
  - launch\_MM\_dispatch, 1137
  - main, 1137
  - run\_with\_field, 1137
- test-fgemm.C, 1138
  - check\_MM, 1138
  - ENABLE\_CHECKER\_fgemm, 1138
  - launch\_MM, 1139
  - launch\_MM\_dispatch, 1139
  - main, 1140
  - run\_with\_field, 1139
- test-fgemv.C, 1140
  - check\_MV, 1140
  - launch\_MV, 1141
  - launch\_MV\_dispatch, 1141
  - main, 1141
  - run\_with\_field, 1141
- test-fger.C, 1142
  - check\_fger, 1142
  - launch\_fger, 1143
  - launch\_fger\_dispatch, 1143
  - main, 1143
  - run\_with\_field, 1143
  - TIME, 1142
- test-fgesv.C, 1144
  - main, 1145
  - run\_with\_field, 1144
  - test\_rect\_fgesv, 1144
  - test\_square\_fgesv, 1144
- test-finit.C, 1145
  - main, 1146
  - run\_with\_field, 1145
  - test\_freduce, 1145
- test-fscal.C, 1146
  - main, 1147
  - test\_fscal, 1146, 1147
  - test\_fscaln, 1147
- test-fsyr2k.C, 1147
  - check\_fsyr2k, 1148
  - ENABLE\_ALL\_CHECKINGS, 1148
  - main, 1148
  - run\_with\_field, 1148
- test-fsyrk.C, 1149
  - check\_computeS1S2, 1150
  - check\_fsyrk, 1149
  - check\_fsyrk\_bkdiag, 1150
  - check\_fsyrk\_diag, 1150
  - ENABLE\_ALL\_CHECKINGS, 1149
  - main, 1150
  - run\_with\_field, 1150
- test-fsytrf.C, 1151
  - main, 1152
  - operator<<, 1151
  - run\_with\_field, 1152
  - test\_generic\_fsytrf, 1151
  - test\_RPM\_fsytrf, 1151
- test-ftrmm.C, 1152
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1153
  - check\_ftrmm, 1153
  - main, 1153
  - run\_with\_field, 1153
- test-ftrmv.C, 1153
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1154
  - check\_ftrmv, 1154
  - ENABLE\_ALL\_CHECKINGS, 1154
  - main, 1154
  - run\_with\_field, 1154
- test-ftrsm-check.C, 1155
  - ENABLE\_ALL\_CHECKINGS, 1155
  - main, 1155
- test-ftrsm.C, 1155
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1156
  - check\_ftrsm, 1156
  - ENABLE\_ALL\_CHECKINGS, 1156
  - main, 1156
  - run\_with\_field, 1156
- test-ftrssyr2k.C, 1157
  - check\_ftrssyr2k, 1157
  - ENABLE\_ALL\_CHECKINGS, 1157
  - main, 1158
  - run\_with\_field, 1157
- test-ftrstr.C, 1158
  - check\_ftrstr, 1158
  - ENABLE\_ALL\_CHECKINGS, 1158
  - main, 1159
  - run\_with\_field, 1159
- test-ftrsv.C, 1159
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1159
  - check\_ftrsv, 1160
  - ENABLE\_ALL\_CHECKINGS, 1159
  - main, 1160
  - run\_with\_field, 1160
- test-ftrtri.C, 1160
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1161
  - check\_ftrtri, 1161
  - ENABLE\_ALL\_CHECKINGS, 1161
  - main, 1161
  - run\_with\_field, 1161
- test-interfaces-c.c, 1161
  - main, 1162
- test-invert-check.C, 1162
  - ENABLE\_ALL\_CHECKINGS, 1162
  - main, 1162
- test-io.C, 1162
  - main, 1163

- run\_with\_field, 1163
- test-lu.C, 1163
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1164
  - \_\_LUDIVINE\_CUTOFF, 1164
  - BASECASE\_K, 1164
  - launch\_test, 1166
  - main, 1167
  - mvcnt, 1167
  - run\_with\_field, 1166
  - tBC, 1167
  - test\_LUdivine, 1164
  - test\_pluq, 1166
  - tgemm, 1167
  - timtot, 1167
  - tperm, 1167
  - trest, 1167
  - ttism, 1167
  - verifPLUQ, 1165
- test-maxdelayeddim.C, 1168
  - main, 1168
  - MAX\_WITH\_SIZE\_T, 1168
  - test, 1168
- test-minpoly.C, 1168
  - check\_minpoly, 1169
  - main, 1169
  - run\_with\_field, 1169
- test-multifile1.C, 1169
- test-multifile2.C, 1169
  - main, 1170
- test-nullspace.C, 1170
  - checkingMessage, 1170
  - main, 1171
  - readOrRandomMatrixWithRankAndRandomRPM, 1170
  - run\_with\_field, 1171
  - test\_nullspace, 1170
- test-permutations.C, 1171
  - checkMonotonicApplyP, 1171
  - main, 1172
  - tBC, 1172
  - tgemm, 1172
  - timtot, 1172
  - tperm, 1172
  - trest, 1172
  - ttism, 1172
- test-pluq-check.C, 1172
  - ENABLE\_ALL\_CHECKINGS, 1173
  - main, 1173
- test-quasisep.C, 1173
  - launch\_test, 1174
  - main, 1174
  - run\_with\_field, 1174
  - test\_BruhatGenerator, 1173
  - testLTQSRPM, 1174
- test-rankprofiles.C, 1174
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1175
  - main, 1175
  - run\_with\_field, 1175
- test-rpm.C, 1175
  - checkRPM, 1176
  - checkSymmetricRPM, 1176
  - main, 1176
- test-simd.C, 1176
  - \_TEST\_ONE, 1177
  - check\_eq, 1178
  - cmp, 1179
  - eval\_func\_on\_array, 1179
  - main, 1179
  - operator<<, 1179
  - TEST\_IMPL, 1178
  - test\_impl, 1179
  - test\_impl\_base, 1179
  - TEST\_ONE\_OP, 1178
  - TEST\_ONE\_OP\_WZ, 1178
- test-solve.C, 1180
  - check\_solve, 1180
  - main, 1180
  - run\_with\_field, 1180
- test-storage-transpose.C, 1180
  - main, 1181
- test-utils.h, 1181
- test\_BruhatGenerator
  - test-quasisep.C, 1173
- test\_colechelon
  - test-echelon.C, 1132
- test\_det
  - test-det.C, 1131
- test\_fadd
  - test-fadd.C, 1134
- test\_faddin
  - test-fadd.C, 1134
- test\_freduce
  - test-finit.C, 1145
- test\_fscal
  - test-fscal.C, 1146, 1147
- test\_fscaln
  - test-fscal.C, 1147
- test\_fsub
  - test-fadd.C, 1135
- test\_fsubin
  - test-fadd.C, 1135
- test\_ftranspose
  - Test< Elt >, 812
- test\_generic\_fsytrf
  - test-fsytrf.C, 1151
- TEST\_IMPL
  - test-simd.C, 1178
- test\_impl
  - test-simd.C, 1179
- test\_impl\_base
  - test-simd.C, 1179
- test\_LUdivine
  - test-lu.C, 1164
- test\_nullspace
  - test-nullspace.C, 1170
- TEST\_ONE\_OP

- test-simd.C, 1178
- TEST\_ONE\_OP\_WZ
  - test-simd.C, 1178
- test\_pluq
  - test-lu.C, 1166
- test\_rect\_fgesv
  - test-fgesv.C, 1144
- test\_redcochelon
  - test-echelon.C, 1133
- test\_redrowechelon
  - test-echelon.C, 1133
- test\_rowechelon
  - test-echelon.C, 1133
- test\_RPM\_fsytrf
  - test-fsytrf.C, 1151
- test\_square\_fgesv
  - test-fgesv.C, 1144
- testLTQSRPM
  - test-quasisep.C, 1174
- TestOneMethod
  - TestOneMethod< Simd >, 814
- TestOneMethod< Simd >, 813
  - Element, 814
  - enable\_if\_t, 814
  - evaluate\_scalar\_method, 814
  - evaluate\_simd\_method, 815
  - getStatus, 815
  - getTestName, 815
  - inputs, 815
  - name, 815
  - nb\_lref, 815
  - outputs\_scalar, 816
  - outputs\_simd, 816
  - TestOneMethod, 814
  - vect\_size, 815
  - vect\_t, 814
  - vectElt, 814
  - writeDebugData, 815
  - writeResultLine, 815
- tfn\_minus, 816
  - operator(), 816
- tfn\_minus\_eq, 816
  - operator(), 816
- tfn\_mul, 817
  - operator(), 817
- tfn\_mul\_eq, 817
  - operator(), 817
- tfn\_plus, 817
  - operator(), 817
- tfn\_plus\_eq, 818
  - operator(), 818
- tgemm
  - test-lu.C, 1167
  - test-permutations.C, 1172
- THREAD\_INDEX
  - parallel.h, 1095
- THREADS
  - benchmark-fgemm-rns.C, 839
- Threads, 818
- threads\_fgemm
  - FFPACK, 385
- threads\_ftsm
  - FFPACK, 385
- THREED
  - benchmark-fgemm-rns.C, 839
- ThreeD, 818
- THREEDA
  - benchmark-fgemm-rns.C, 839
- ThreeDAdaptive, 818
- ThreeDInPlace, 818
- THREEDIP
  - benchmark-fgemm-rns.C, 839
- TIME
  - test-fger.C, 1142
- TIME\_CHECKER\_CHARPOLY
  - test-charpoly-check.C, 1128
- TIME\_CHECKER\_Det
  - test-det-check.C, 1130
- Timer
  - FFLAS, 78
- timer.h, 1182
- timtot
  - test-lu.C, 1167
  - test-permutations.C, 1172
- TInverter
  - FFPACK, 366, 369
- tmain
  - benchmark-fgemm-mp.C, 837
- tperm
  - test-lu.C, 1167
  - test-permutations.C, 1172
- transpose
  - BlockTransposeSIMD< Field, Simd, >, 436
  - Simd128\_impl< true, true, false, 2 >, 607
  - Simd128\_impl< true, true, false, 4 >, 617
  - Simd128\_impl< true, true, false, 8 >, 627
  - Simd128\_impl< true, true, true, 2 >, 636
  - Simd128\_impl< true, true, true, 4 >, 646
  - Simd128\_impl< true, true, true, 8 >, 656
  - Simd256\_impl< true, false, true, 8 >, 667
  - Simd256\_impl< true, true, false, 2 >, 679
  - Simd256\_impl< true, true, false, 4 >, 695
  - Simd256\_impl< true, true, false, 8 >, 708
  - Simd256\_impl< true, true, true, 2 >, 716
  - Simd256\_impl< true, true, true, 4 >, 728, 735
  - Simd256\_impl< true, true, true, 8 >, 746
  - Simd512\_impl< true, false, true, 8 >, 756
  - Simd512\_impl< true, true, false, 8 >, 767
  - Simd512\_impl< true, true, true, 8 >, 776
- trest
  - test-lu.C, 1167
  - test-permutations.C, 1172
- trinv\_left
  - FFPACK, 329, 388
- trinv\_left\_modular\_double
  - ffpack.C, 1002

- ffpack\_c.h, [1037](#)
- TRSMBound
  - FFLAS::Protected, [227](#)
- TRSMHelper
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [819](#)
- TRSMHelper< ReclterTrait, ParSeqTrait >, [819](#)
  - parseq, [820](#)
  - pMMH, [819](#), [820](#)
  - TRSMHelper, [819](#)
- TTimer
  - arithprog.C, [827](#)
  - autotune/charpoly.C, [859](#)
  - autotune/pluq.C, [1103](#)
  - benchmark-dgemm.C, [831](#)
  - benchmark-dgetrf.C, [832](#)
  - benchmark-dgetri.C, [833](#)
  - benchmark-dsytrf.C, [834](#)
  - benchmark-dtrsm.C, [834](#)
  - benchmark-dtrtri.C, [835](#)
  - fsyrk.C, [1083](#)
  - fsytrf.C, [1084](#)
  - fttrtri.C, [1085](#)
  - winograd.C, [1183](#)
- ttrsm
  - test-lu.C, [1167](#)
  - test-permutations.C, [1172](#)
- TWOD
  - benchmark-fgemm-rns.C, [839](#)
- TwoD, [820](#)
- TWODA
  - benchmark-fgemm-rns.C, [839](#)
- TwoDAdaptive, [820](#)
- type
  - Argument, [430](#)
  - associatedDelayedField< const FFPACK::RNSIntegerMod>  
RNS >, [431](#)
  - associatedDelayedField< const Givaro::Modular<  
T, X >>, [431](#)
  - associatedDelayedField< const Givaro::ModularBalanced<  
T >>, [432](#)
  - associatedDelayedField< const Givaro::ZRing< T  
>>, [432](#)
  - associatedDelayedField< Field >, [430](#)
  - CompactElement< double >, [447](#)
  - CompactElement< Element >, [447](#)
  - CompactElement< float >, [447](#)
  - CompactElement< int16\_t >, [448](#)
  - CompactElement< int32\_t >, [448](#)
  - CompactElement< int64\_t >, [448](#)
  - is\_simd< T >, [511](#)
- TYPE\_BOOL
  - args-parser.h, [825](#)
- TYPE\_DOUBLE
  - args-parser.h, [826](#)
- TYPE\_INT
  - args-parser.h, [826](#)
- TYPE\_INTEGER
  - args-parser.h, [826](#)
- type\_integer
  - args-parser.h, [826](#)
- TYPE\_INTLIST
  - args-parser.h, [826](#)
- TYPE\_LONGLONG
  - args-parser.h, [826](#)
- TYPE\_NONE
  - args-parser.h, [826](#)
- TYPE\_STR
  - args-parser.h, [826](#)
- type\_string
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [613](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [686](#), [689](#)
  - Simd256\_impl< true, true, false, 8 >, [704](#)
  - Simd256\_impl< true, true, true, 2 >, [714](#)
  - Simd256\_impl< true, true, true, 4 >, [726](#), [732](#)
  - Simd256\_impl< true, true, true, 8 >, [743](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [773](#)
- TYPE\_UINT64
  - args-parser.h, [826](#)
- unfit
  - FFLAS::Protected, [236](#)
- unpackhi
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [694](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [735](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [766](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- unpackhi\_intrinsic
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)

- Simd256\_impl< true, true, false, 2 >, [678](#)
- Simd256\_impl< true, true, false, 4 >, [693](#), [694](#)
- Simd256\_impl< true, true, false, 8 >, [707](#)
- Simd256\_impl< true, true, true, 2 >, [715](#)
- Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
- Simd256\_impl< true, true, true, 8 >, [745](#)
- Simd512\_impl< true, false, true, 8 >, [755](#)
- Simd512\_impl< true, true, false, 8 >, [766](#)
- Simd512\_impl< true, true, true, 8 >, [775](#)
- unpacklo
  - ScalFunctions< Element >, [593](#)
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [694](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [766](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- unpacklo\_intrinsic
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [666](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [693](#)
  - Simd256\_impl< true, true, false, 8 >, [707](#)
  - Simd256\_impl< true, true, true, 2 >, [715](#)
  - Simd256\_impl< true, true, true, 4 >, [727](#), [734](#)
  - Simd256\_impl< true, true, true, 8 >, [745](#)
  - Simd512\_impl< true, false, true, 8 >, [755](#)
  - Simd512\_impl< true, true, false, 8 >, [766](#)
  - Simd512\_impl< true, true, true, 8 >, [775](#)
- unpacklohi
  - ScalFunctions< Element >, [594](#)
  - Simd128\_impl< true, true, false, 2 >, [606](#)
  - Simd128\_impl< true, true, false, 4 >, [616](#)
  - Simd128\_impl< true, true, false, 8 >, [627](#)
  - Simd128\_impl< true, true, true, 2 >, [635](#)
  - Simd128\_impl< true, true, true, 4 >, [645](#)
  - Simd128\_impl< true, true, true, 8 >, [655](#)
  - Simd256\_impl< true, false, true, 8 >, [667](#)
  - Simd256\_impl< true, true, false, 2 >, [678](#)
  - Simd256\_impl< true, true, false, 4 >, [694](#)
  - Simd256\_impl< true, true, false, 8 >, [708](#)
  - Simd256\_impl< true, true, true, 2 >, [716](#)
  - Simd256\_impl< true, true, true, 4 >, [728](#), [735](#)
- Simd256\_impl< true, true, true, 8 >, [745](#)
- Simd512\_impl< true, false, true, 8 >, [755](#)
- Simd512\_impl< true, true, false, 8 >, [766](#)
- Simd512\_impl< true, true, true, 8 >, [775](#)
- UnparametricTag, [820](#)
- updateD
  - FFPACK::Protected, [422](#)
- USE\_OPENMP
  - config.h, [877](#)
- UserTimer
  - FFLAS, [78](#)
- utils.h, [1182](#)
- v
  - Simd128\_impl< true, true, false, 2 >::Converter, [451](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [451](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [451](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [452](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [452](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [453](#)
  - Simd256\_impl< true, false, true, 8 >::Converter, [453](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [453](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [454](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [454](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [455](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [455](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [455](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [456](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [456](#)
- val
  - Coo< Field >, [460](#)
  - Coo< ValT, IdxT >, [458](#), [461](#)
- valid
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [605](#)
  - Simd128\_impl< true, true, false, 4 >, [615](#)
  - Simd128\_impl< true, true, false, 8 >, [626](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [677](#)
  - Simd256\_impl< true, true, false, 4 >, [692](#)
  - Simd256\_impl< true, true, false, 8 >, [706](#)

- Simd256\_impl< true, true, true, 2 >, [714](#)
- Simd256\_impl< true, true, true, 4 >, [726](#), [732](#)
- Simd256\_impl< true, true, true, 8 >, [743](#)
- Simd512\_impl< true, false, true, 8 >, [753](#)
- Simd512\_impl< true, true, false, 8 >, [765](#)
- Simd512\_impl< true, true, true, 8 >, [773](#)
- VALUE
  - parallel.h, [1096](#)
- value
  - AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, [427](#)
  - AlgoChooser< ModeT, ParSeq >, [427](#)
  - ALL< false, v... >, [428](#)
  - ALL< true, v... >, [428](#)
  - ALL<>, [428](#)
  - AreEqual< X, X >, [429](#)
  - AreEqual< X, Y >, [429](#)
  - compatible\_data\_type< Field >, [448](#)
  - compatible\_data\_type< Givaro::ZRing< double > >, [449](#)
  - compatible\_data\_type< Givaro::ZRing< float > >, [449](#)
  - ElementTraits< double >, [466](#)
  - ElementTraits< Element >, [466](#)
  - ElementTraits< FFPACK::rns\_double\_elt >, [466](#)
  - ElementTraits< float >, [467](#)
  - ElementTraits< Givaro::Integer >, [467](#)
  - ElementTraits< int16\_t >, [467](#)
  - ElementTraits< int32\_t >, [468](#)
  - ElementTraits< int64\_t >, [468](#)
  - ElementTraits< int8\_t >, [468](#)
  - ElementTraits< Reclnt::rint< K > >, [469](#)
  - ElementTraits< Reclnt::rmint< K, MG > >, [469](#)
  - ElementTraits< Reclnt::ruint< K > >, [469](#)
  - ElementTraits< uint16\_t >, [470](#)
  - ElementTraits< uint32\_t >, [470](#)
  - ElementTraits< uint64\_t >, [470](#)
  - ElementTraits< uint8\_t >, [471](#)
  - has\_minus\_eq\_impl< C >, [500](#)
  - has\_minus\_impl< C >, [501](#)
  - has\_mul\_eq\_impl< C >, [501](#)
  - has\_mul\_impl< C >, [501](#)
  - has\_operation< T >, [502](#)
  - has\_plus\_eq\_impl< C >, [502](#)
  - has\_plus\_impl< C >, [503](#)
  - is\_all\_same< T, Args... >, [510](#)
  - is\_all\_same<>, [510](#)
  - is\_simd< T >, [511](#)
  - ModeTraits< Field >, [545](#)
  - ModeTraits< Givaro::Modular< Element, Compute > >, [545](#)
  - ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, [545](#)
  - ModeTraits< Givaro::Modular< int16\_t, Compute > >, [546](#)
  - ModeTraits< Givaro::Modular< int32\_t, Compute > >, [546](#)
  - ModeTraits< Givaro::Modular< int64\_t, uint64\_t > >, [547](#)
  - ModeTraits< Givaro::Modular< int8\_t, Compute > >, [547](#)
  - ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >, [547](#)
  - ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [548](#)
  - ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [548](#)
  - ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [548](#)
  - ModeTraits< Givaro::ModularBalanced< Element > >, [549](#)
  - ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [549](#)
  - ModeTraits< Givaro::ModularBalanced< int16\_t > >, [549](#)
  - ModeTraits< Givaro::ModularBalanced< int32\_t > >, [550](#)
  - ModeTraits< Givaro::ModularBalanced< int8\_t > >, [550](#)
  - ModeTraits< Givaro::Montgomery< T > >, [551](#)
  - ModeTraits< Givaro::ZRing< double > >, [551](#)
  - ModeTraits< Givaro::ZRing< float > >, [551](#)
  - ModeTraits< Givaro::ZRing< Givaro::Integer > >, [552](#)
  - need\_field\_characteristic< Field >, [552](#)
  - need\_field\_characteristic< Givaro::Modular< Field > >, [553](#)
  - need\_field\_characteristic< Givaro::ModularBalanced< Field > >, [553](#)
  - SimdChooser< T, false, b >, [782](#)
  - SimdChooser< T, true, false >, [783](#)
  - SimdChooser< T, true, true >, [783](#)
  - width< double >, [821](#)
  - width< float >, [821](#)
  - width< T >, [821](#)
- vand
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [609](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd128i\_base, [662](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 4 >, [700](#)
  - Simd256\_impl< true, true, true, 4 >, [740](#)
  - Simd512\_impl< true, true, false, 8 >, [770](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
  - Simd512i\_base, [782](#)
- vandnot
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [610](#)
  - Simd128\_impl< true, true, false, 4 >, [620](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)

- Simd128\_impl< true, true, true, 2 >, [640](#)
- Simd128\_impl< true, true, true, 4 >, [650](#)
- Simd128\_impl< true, true, true, 8 >, [661](#)
- Simd128i\_base, [662](#)
- Simd256\_impl< true, false, true, 8 >, [670](#)
- Simd256\_impl< true, true, false, 4 >, [701](#)
- Simd256\_impl< true, true, true, 4 >, [740](#)
- Simd512\_impl< true, true, false, 8 >, [770](#)
- Simd512\_impl< true, true, true, 8 >, [780](#)
- Simd512i\_base, [782](#)
- VEC\_ADD
  - FFLAS::vectorised, [298](#)
- VEC\_SUB
  - FFLAS::vectorised, [298](#)
- vect\_size
  - FieldSimd< \_Field >, [480](#)
  - NoSimd< T >, [554](#)
  - Simd128\_impl< true, true, false, 2 >, [610](#)
  - Simd128\_impl< true, true, false, 4 >, [620](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [661](#)
  - Simd256\_impl< true, false, true, 8 >, [671](#)
  - Simd256\_impl< true, true, false, 2 >, [681](#)
  - Simd256\_impl< true, true, false, 4 >, [701](#)
  - Simd256\_impl< true, true, false, 8 >, [711](#)
  - Simd256\_impl< true, true, true, 2 >, [721](#)
  - Simd256\_impl< true, true, true, 4 >, [740](#)
  - Simd256\_impl< true, true, true, 8 >, [750](#)
  - Simd512\_impl< true, false, true, 8 >, [759](#)
  - Simd512\_impl< true, true, false, 8 >, [770](#)
  - Simd512\_impl< true, true, true, 8 >, [781](#)
  - TestOneMethod< Simd >, [815](#)
- vect\_t
  - FieldSimd< \_Field >, [476](#)
  - NoSimd< T >, [553](#)
  - Simd128\_impl< true, true, false, 2 >, [602](#)
  - Simd128\_impl< true, true, false, 4 >, [612](#)
  - Simd128\_impl< true, true, false, 8 >, [623](#)
  - Simd128\_impl< true, true, true, 2 >, [633](#)
  - Simd128\_impl< true, true, true, 4 >, [643](#)
  - Simd128\_impl< true, true, true, 8 >, [653](#)
  - simd128\_int64.inl, [1119](#)
  - Simd128i\_base, [662](#)
  - Simd256\_impl< true, false, true, 8 >, [665](#)
  - Simd256\_impl< true, true, false, 2 >, [674](#)
  - Simd256\_impl< true, true, false, 4 >, [686](#)
  - Simd256\_impl< true, true, false, 8 >, [703](#)
  - Simd256\_impl< true, true, true, 2 >, [713](#)
  - Simd256\_impl< true, true, true, 4 >, [724](#), [725](#)
  - Simd256\_impl< true, true, true, 8 >, [742](#)
  - simd256\_int64.inl, [1122](#)
  - Simd256i\_base, [751](#)
  - Simd512\_impl< true, false, true, 8 >, [753](#)
  - Simd512\_impl< true, true, false, 8 >, [762](#)
  - Simd512\_impl< true, true, true, 8 >, [772](#)
  - simd512\_int64.inl, [1125](#)
  - Simd512i\_base, [781](#)
  - TestOneMethod< Simd >, [814](#)
- vectElt
  - ScalFunctions< Element >, [590](#)
  - TestOneMethod< Simd >, [814](#)
- verification\_PLUQ
  - benchmark-pluq.C, [852](#)
- verifPLUQ
  - test-lu.C, [1165](#)
- VERSION
  - config.h, [877](#)
- vor
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [610](#)
  - Simd128\_impl< true, true, false, 4 >, [619](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [660](#)
  - Simd128i\_base, [662](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 4 >, [700](#)
  - Simd256\_impl< true, true, true, 4 >, [740](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
  - Simd512i\_base, [782](#)
- vxor
  - ScalFunctions< Element >, [591](#)
  - Simd128\_impl< true, true, false, 2 >, [610](#)
  - Simd128\_impl< true, true, false, 4 >, [620](#)
  - Simd128\_impl< true, true, false, 8 >, [630](#)
  - Simd128\_impl< true, true, true, 2 >, [640](#)
  - Simd128\_impl< true, true, true, 4 >, [650](#)
  - Simd128\_impl< true, true, true, 8 >, [661](#)
  - Simd128i\_base, [662](#)
  - Simd256\_impl< true, false, true, 8 >, [670](#)
  - Simd256\_impl< true, true, false, 4 >, [700](#)
  - Simd256\_impl< true, true, true, 4 >, [740](#)
  - Simd512\_impl< true, true, false, 8 >, [769](#)
  - Simd512\_impl< true, true, true, 8 >, [780](#)
  - Simd512i\_base, [782](#)
- WAIT
  - parallel.h, [1095](#)
- width< double >, [821](#)
  - value, [821](#)
- width< float >, [821](#)
  - value, [821](#)
- width< T >, [820](#)
  - value, [821](#)
- Winograd, [821](#)
  - FFLAS::BLAS3, [206](#)
- winograd.C, [1182](#)
  - balanced, [1183](#)
  - DOUBLE\_TO\_FLOAT\_CROSSOVER, [1183](#)
  - GFOPS, [1183](#)
  - main, [1183](#)
  - TTimer, [1183](#)
- Winograd\_L\_S

- FFLAS::BLAS3, [210](#)
- Winograd\_LR\_S
  - FFLAS::BLAS3, [210](#)
- Winograd\_R\_S
  - FFLAS::BLAS3, [210](#)
- WinogradAcc\_2\_24
  - FFLAS::BLAS3, [208](#)
- WinogradAcc\_2\_27
  - FFLAS::BLAS3, [208](#)
- WinogradAcc\_3\_21
  - FFLAS::BLAS3, [207](#)
- WinogradAcc\_3\_23
  - FFLAS::BLAS3, [207](#)
- WinogradAcc\_L\_S
  - FFLAS::BLAS3, [209](#)
- WinogradAcc\_LR
  - FFLAS::BLAS3, [208](#)
- WinogradAcc\_R\_S
  - FFLAS::BLAS3, [209](#)
- WinogradCalc
  - FFLAS::Protected, [229](#)
- WinogradPar, [821](#)
- WinogradSteps
  - FFLAS::Protected, [228](#)
- WinogradThreshold
  - FFLAS::Protected, [227](#), [228](#)
- WinoPar
  - FFLAS::BLAS3, [206](#)
- WINOTHRESHOLD
  - fflas.h, [907](#)
- WRITE
  - parallel.h, [1096](#)
- write
  - RNSInteger< RNS >, [581](#)
  - RNSIntegerMod< RNS >, [586](#)
- write\_field
  - Matio.h, [1093](#)
- write\_matrix
  - benchmark-fgemv-mp.C, [841](#)
  - RNSIntegerMod< RNS >, [586](#)
- write\_matrix\_long
  - RNSIntegerMod< RNS >, [587](#)
- writeCommandString
  - FFLAS, [198](#)
- writeDebugData
  - TestOneMethod< Simd >, [815](#)
- writeDnsFormat
  - FFLAS, [161](#)
- WriteMatrix
  - FFLAS, [198](#), [200](#)
- WritePermutation
  - FFLAS, [200](#)
- writeResultLine
  - TestOneMethod< Simd >, [815](#)
- zero
  - FFLAS, [81](#)
  - FieldSimd< \_Field >, [478](#)
  - RNSInteger< RNS >, [581](#)
- RNSIntegerMod< RNS >, [588](#)
- ScalFunctions< Element >, [591](#)
- Simd128\_impl< true, true, false, 2 >, [609](#)
- Simd128\_impl< true, true, false, 4 >, [619](#)
- Simd128\_impl< true, true, false, 8 >, [630](#)
- Simd128\_impl< true, true, true, 2 >, [640](#)
- Simd128\_impl< true, true, true, 4 >, [650](#)
- Simd128\_impl< true, true, true, 8 >, [660](#)
- Simd128i\_base, [662](#)
- Simd256\_impl< true, false, true, 8 >, [665](#)
- Simd256\_impl< true, true, false, 2 >, [681](#)
- Simd256\_impl< true, true, false, 4 >, [700](#)
- Simd256\_impl< true, true, false, 8 >, [711](#)
- Simd256\_impl< true, true, true, 2 >, [721](#)
- Simd256\_impl< true, true, true, 4 >, [739](#)
- Simd256\_impl< true, true, true, 8 >, [750](#)
- Simd256i\_base, [751](#)
- Simd512\_impl< true, false, true, 8 >, [753](#)
- Simd512\_impl< true, true, false, 8 >, [769](#)
- Simd512\_impl< true, true, true, 8 >, [780](#)
- Simd512i\_base, [781](#)
- ZOSparseMatrix
  - FFLAS, [76](#)