| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [ConversionEngine/src](http://docs.google.com/index.html) - ConversionEngine.cpp (source / [functions](http://docs.google.com/ConversionEngine.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 482 | 495 | 97.4 % | | Date: | 2012-03-23 |  | Functions: | 10 | 12 | 83.3 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 59 | 90 | 65.6 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include <iostream>  2 : : #include <boost/bind.hpp>  3 : : #include <boost/mem\_fn.hpp>  4 : : #include <boost/ref.hpp>  5 : : #include <sys/time.h>  6 : :   7 : : #include "ArbitraryModeCellProcessorList.h"  8 : : #include "ArbitraryModeProcessor.h"  9 : : #include "ConversionEngine.h"  10 : : #include "ConverterParams.h"  11 : : #include "HealthCheckCounter.h"  12 : : #include "MemMgr.h"  13 : : #include "PatternProcessor.h"  14 : : #include "print.h"  15 : : #include "SFTableCreator.h"  16 : : #include "Sdf12iTypes.h"  17 : : #include "UtilTypes.h"  18 : : #include "Logger.h"  19 : : #include "SDDB.h"  20 : : #include "Version.h"  21 : :   22 : : namespace nft  23 : : {  24 : :   25 : : void  26 : 1095 : ConversionEngine::convert(DPL::LOG\_MODE logLvl) throw(DPL::Error \*) {  27 : : //ConversionEngine::convert() throw(DPL::Error \*) {  28 : 1095 : PatternEdgeVectors::initRepo();  29 : 1095 : HealthCheckCounter::instance()->setDiagLevel(  30 : : HealthCheckCounter::eUpdateNever );  31 : 1095 : Logger::instance()->log(LogMessageTable::eCONV\_START,  32 : : gVERSION  33 : : , ConverterParams::instance()->getHeader()->uiColumnNo  34 : : , ConverterParams::instance()->getHeader()->uiStripeNo  35 : : , ConverterParams::instance()->getHeader()->uiDpbNo);  36 : : //Eupa >>  37 [ + + ]: 1095 : if(2 == logLvl || 3 == logLvl)  38 : : {  39 : 1063 : dumpConvConfParams();  40 : : }  41 : : else  42 : : {  43 : : //No dumping for loglvl = 1  44 : : }  45 : : //<< Eupa  46 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  47 : 1095 : = mParams->getStartupMessage();  48 : 1095 : ShortTimeInfo::setNumRepetitions( pStartMsg->uiRepCount );  49 : 1095 : SDDB::instance()->setLevel( static\_cast<sd::SDLevel>(pStartMsg->usDiag) );  50 : 1095 : MemMgr::instance()->setOffset( sizeof(SimplePattern) >> 1 ); //Perf  51 [ + + ][ + + ]: 1095 : switch( pStartMsg->usProcessingMode )  [ - + ][ + + ]  [ + + ]  52 : : {  53 : : case eProcessNormal:  54 : : case eProcessTFLess:  55 : : {  56 : 1048 : try  57 : : {  58 : 1048 : convertFixedMode();  59 : : }  60 [ + - ]: 62 : catch(DPL::Error \*pChild)  61 : : {  62 : 62 : Logger::instance()->log(LogMessageTable::eCONV\_ERROR,  63 : : "Conversion to Fixed mode failed");  64 : 62 : ErrorHandler \*pErr = ErrorHandler::instance();  65 : 62 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  66 : : ErrorHandler::eErrFunction,  67 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  68 : : "ConversionEngine::convertFixedMode"));  69 : : }  70 : 0 : break;  71 : : }  72 : : case eProcessArbitrary:  73 : : case eProcessArbitraryTFLess:  74 : : {  75 : 46 : try  76 : : {  77 : 46 : convertArbitraryMode();  78 : : }  79 [ + - ]: 2 : catch(DPL::Error \*pChild)  80 : : {  81 : 2 : Logger::instance()->log(LogMessageTable::eCONV\_ERROR,  82 : : "Conversion to Arbitrary mode failed");  83 : 2 : ErrorHandler \*pErr = ErrorHandler::instance();  84 : 2 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  85 : : ErrorHandler::eErrFunction,  86 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  87 : : "ConversionEngine::convertArbitraryMode"));  88 : : }  89 : 0 : break;  90 : : }  91 : : default:  92 : : {  93 : 1 : e9k\_error("This flow is not implemented\n");  94 : : }   95 : : }  96 [ + - ]: 1095 : Logger::instance()->log(LogMessageTable::eCONV\_END,  97 : : "Conversion process ends.");  98 : : }  99 : :   100 : : bool  101 : 7 : ConversionEngine::setLastStripe() throw(DPL::Error \*) {  102 [ # # ]: 7 : return mOutputGen->setLastStripeFlag();  103 : : }  104 : :   105 : : void  106 : 1031 : ConversionEngine::getOutputMemory( MemInfo &outMemory ) {  107 [ + + ]: 1031 : if( mOutputGen )  108 : : {  109 : 1031 : outMemory = mOutputGen->getOutputMemory();  110 : : }  111 : : }  112 : :   113 : : void  114 : 1048 : ConversionEngine::convertFixedMode() throw(DPL::Error \*) {  115 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  116 : 1048 : = mParams->getStartupMessage();  117 : : //1. Create SF Table  118 : 1048 : try  119 : : {  120 : 1048 : mFixedModeSFTable = SFTableCreator::constructSFTable(  121 : : mParams );  122 : : }  123 [ # # ]: 0 : catch(DPL::Error \*pChild)  124 : : {  125 : 0 : Logger::instance()->log(LogMessageTable::eCONV\_ERROR,  126 : : "Construction of SF Table failed");  127 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  128 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  129 : : ErrorHandler::eErrFunction,  130 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  131 : : "SFTableCreator::constructSFTable"));  132 : : }  133 : :   134 : : ProcessingMode procMode = static\_cast<ProcessingMode>(  135 : 1048 : pStartMsg->usProcessingMode);  136 : :   137 : : //1.a Register TF distributor to all nodes  138 [ + + ]: 1048 : if( eProcessTFLess != procMode )  139 : : {  140 : 1020 : mFixedModeTFDistributor = new TFDistributor(  141 : : mFixedModeSFTable);  142 : 1020 : uint32\_t numSFL = pStartMsg->uiSFLCountInDpb;  143 : 1020 : SFLayer::Pointer layers = mFixedModeSFTable->getSFLayers();  144 [ + - ][ + + ]: 2096 : for( SFLayer::Pointer layer = layers;  145 : : layer != layers + numSFL; layer++ )  146 : : {  147 : 1076 : layer->getTFL()->registerTFDistributionHandler(  148 : 1076 : boost::ref(\*mFixedModeTFDistributor) );  149 : : }  150 : : }  151 : :   152 : : //2. Create window  153 : 1048 : mWindow = new SlidingWindow( cSlidingWindowThreadID, mFixedModeSFTable );  154 : : //3. Create cell processors  155 : : PatternDivisionMode divMode = static\_cast<PatternDivisionMode>(  156 : 1048 : mParams->getColumnParams()->div.mode);  157 : :   158 : 1048 : setupShotTimeCalc();  159 : :   160 : 1048 : const sgs\_prox\_param\_t &proxParams = mParams->getColumnParams()->prox;  161 : : ShotTimeCalculator::ProxMode proxMode  162 : : = static\_cast<ShotTimeCalculator::ProxMode>(  163 : 1048 : proxParams.mode & 0x00000001);  164 : : ShotTimeCalculator calc( proxMode, mParams->getDoseTable(),  165 : 1048 : mParams->getKBRTable(), mParams->getDOCTable() );  166 : 1048 : uint16\_t numThreads = pStartMsg->usConcurrency;  167 [ + - + + ]: 2144 : for( ThreadID thread = 0; thread < numThreads; thread++ )  168 : : {  169 : : FixedModeShotDistributor::Pointer distributor  170 : : = new FixedModeShotDistributor(  171 : : calc,  172 : : mFixedModeSFTable, procMode,  173 : 1096 : mParams);  174 : 1096 : mFixedModeDistributors.push\_back( distributor );  175 : : DivisionParams \*divParam = new DivisionParams(  176 : : mParams->getSlopeData(),  177 : 1096 : mParams->getDIVTable());  178 : 1096 : PatternProcessor \*patProc = new PatternProcessor();  179 : 1096 : patProc->setThreadID( thread );  180 : 1096 : patProc->setMode( divMode );  181 : 1096 : patProc->setDivisionParams( divParam );  182 : 1096 : patProc->registerDistributor( boost::ref(\*distributor) );  183 : : FixedModeCellProcessor::Pointer cellProc  184 : : = new FixedModeCellProcessor( thread,  185 : 1096 : mParams, patProc);  186 : 1096 : cellProc->setWindow( mWindow );  187 : 1096 : mFixedModeCellProcs.push\_back( std::make\_pair(cellProc,false) );  188 : 1096 : mThreadManager.startThread( cellProc );  189 : 1096 : mWindow->addCellProcessor( cellProc );  190 : : }  191 : :   192 : 1048 : bool isTFLessMode = pStartMsg->usProcessingMode & 0x02;  193 : : //TODO::ANALYZE  194 : 1048 : size\_t estShotCount = pStartMsg->ullEstimatedShotCount;  195 : : size\_t estTFcount = ( isTFLessMode )  196 [ + + ]: 1048 : ? estShotCount : estShotCount >> 4;  197 [ - + ]: 1048 : estTFcount = (0 == estTFcount) ? 1:estTFcount;  198 : 1048 : size\_t estSFcount = estTFcount >> 8;  199 [ + - ]: 1048 : estSFcount = (0 == estSFcount) ? 1:estSFcount;  200 : : size\_t projectedDataSize = estShotCount \* sizeof( SDF12iShot )  201 : : + estTFcount \* sizeof( SDF12iTFHeader )  202 : 1048 : + estSFcount \* sizeof( SDF12iSFHeader );  203 : : ShotTimeCalculator timeCalc(  204 : : proxMode,  205 : : mParams->getDoseTable(),  206 : : mParams->getKBRTable(),  207 : 1048 : mParams->getDOCTable() );  208 : :   209 : : //We assume that each SF has around 32 shots  210 : 1048 : mOutputGen = new OutputGenerator( cOutputGenThreadID,  211 : : true, projectedDataSize,  212 : : mBlkUnit, mShotTimeStep, mShotTimeStep, timeCalc);  213 : 1048 : mOutputGen->setFixedModeSFTable( mFixedModeSFTable );  214 : 1048 : mWindow->setOutputGenerator( mOutputGen );  215 : 1048 : mThreadManager.startThread( mOutputGen );  216 : 1048 : struct timeval startTime;  217 : 1048 : gettimeofday( &startTime, NULL );  218 : 1048 : mThreadManager.startThread( mWindow );  219 : : //All threads started .. now wait for them to complete  220 : 1048 : mThreadManager.waitForCompletion();  221 : 1048 : struct timeval endTime;  222 : 1048 : gettimeofday( &endTime, NULL );  223 : :   224 : : long long usecs = ( endTime.tv\_sec - startTime.tv\_sec) \* 1000000  225 : 1048 : + ( endTime.tv\_usec - startTime.tv\_usec);  226 : 1048 : e9k\_info("Conversion time: %lld sec %lld usec\n",  227 : : usecs / 1000000,  228 : : usecs % 1000000  229 : : );  230 : :   231 : 1048 : SDDB \*pSDDB = SDDB::instance();  232 : 1048 : try  233 : : {  234 : 1048 : pSDDB->conversionCompleted();  235 : : }  236 [ # # ]: 0 : catch( DPL::Error \*pChild ) {  237 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  238 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  239 : : ErrorHandler::eErrFunction,  240 : : pErr->getErrorMsg(  241 : : ErrorHandler::eErrFunction,  242 : : "SDDB::conversionCompleted"));  243 : : }  244 : :   245 : 2144 : for( FixedModeCellProcessorList::iterator it = mFixedModeCellProcs.begin();  246 : : it != mFixedModeCellProcs.end(); ++it )  247 : : {  248 : 1096 : FixedModeCellProcessor::Pointer cellProc = (\*it).first;  249 [ + - ]: 1096 : delete cellProc;  250 : : }  251 : 1048 : for( std::vector<FixedModeShotDistributor::Pointer>::iterator it  252 : 2144 : = mFixedModeDistributors.begin();  253 : : it != mFixedModeDistributors.end(); ++it )  254 : : {  255 : 1096 : delete (\*it);  256 : : }  257 : 1048 : mFixedModeDistributors.clear();  258 : 1048 : mFixedModeCellProcs.clear();  259 : 1048 : uint32\_t numSFL = pStartMsg->uiSFLCountInDpb;  260 : 1048 : SFLayer::Pointer layers = mFixedModeSFTable->getSFLayers();  261 [ + - ][ + + ]: 2152 : for( uint32\_t i = 0; i < numSFL; i++ )  262 : : {  263 : 1104 : SFLayer::Pointer layer = layers + i;  264 : 1104 : layer->releaseSFGroups();  265 : 1104 : TFLayers::Pointer tfl = layer->getTFL();  266 [ + + ]: 1104 : if( tfl )  267 : : {  268 : 1076 : tfl->releaseTFLayers();  269 : : }  270 : : }  271 [ + - ][ + + ]: 2152 : delete []layers;  272 [ + - ]: 1048 : delete mFixedModeSFTable;  273 [ + - ]: 1048 : delete mWindow;  274 [ + + ]: 1048 : delete mFixedModeTFDistributor;  275 : :   276 [ + + ]: 1048 : if( DPL::Error \*pErrObj = mThreadManager.getErrorObject() )  277 : : {  278 : 62 : ErrorHandler \*pErr = ErrorHandler::instance();  279 : 62 : pErr->errorAdd(pErrObj, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  280 : : ErrorHandler::eErrFunction,  281 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  282 : : "ThreadManager::startThread"));  283 : : }  284 : : }  285 : :   286 : : void  287 : 46 : ConversionEngine::convertArbitraryMode() throw(DPL::Error \*) {  288 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  289 : 46 : = mParams->getStartupMessage();  290 : 46 : setupShotTimeCalc();  291 : :   292 : 46 : ArbitraryModeCellProcessorList cellProcs;  293 : :   294 : : ProcessingMode procMode = static\_cast<ProcessingMode>(  295 : 46 : pStartMsg->usProcessingMode);  296 : 46 : size\_t estShotCount = pStartMsg->ullEstimatedShotCount;  297 : 46 : bool isTFLessMode = pStartMsg->usProcessingMode & 0x02;  298 : : size\_t estTFcount = ( isTFLessMode )  299 [ + + ]: 46 : ? estShotCount : estShotCount >> 5;  300 [ - + ]: 46 : estTFcount = (0 == estTFcount) ? 1:estTFcount;  301 : 46 : size\_t estSFcount = estTFcount >> 10;  302 [ + - ]: 46 : estSFcount = (0 == estSFcount) ? 1:estSFcount;  303 : : size\_t projectedDataSize = estShotCount \* sizeof( SDF12iShot )  304 : : + estTFcount \* sizeof( SDF12iTFHeader )  305 : 46 : + estSFcount \* sizeof( SDF12iSFHeader );  306 : 46 : const sgs\_prox\_param\_t &proxParams = mParams->getColumnParams()->prox;  307 : : ShotTimeCalculator::ProxMode proxMode  308 : : = static\_cast<ShotTimeCalculator::ProxMode>(  309 : 46 : proxParams.mode & 0x00000001);  310 : : ShotTimeCalculator calc( proxMode, mParams->getDoseTable(),  311 : 46 : mParams->getKBRTable(), mParams->getDOCTable() );  312 : :   313 : : ShotTimeCalculator timeCalc(  314 : : proxMode,  315 : : mParams->getDoseTable(),  316 : : mParams->getKBRTable(),  317 : 46 : mParams->getDOCTable() );  318 : :   319 : : //We assume that each SF has around 32 shots  320 : 46 : mOutputGen = new OutputGenerator( cOutputGenThreadID,  321 : : false, projectedDataSize,  322 : : mBlkUnit, mShotTimeStep, mShotTimeStep, timeCalc);  323 : :   324 : 46 : std::vector<ArbitraryModeShotDistributor::Pointer> distributors;  325 : : ArbitraryModeProcessor \*processor  326 : : = new ArbitraryModeProcessor(  327 : 46 : cSlidingWindowThreadID, mOutputGen );  328 : :   329 : : PatternDivisionMode divMode = static\_cast<PatternDivisionMode>(  330 : 46 : mParams->getColumnParams()->div.mode);  331 : :   332 : 46 : uint16\_t numThreads = pStartMsg->usConcurrency;  333 [ + - ][ - + ]: 92 : for( ThreadID thread = 0; thread < numThreads; thread++ )  334 : : {  335 : : ArbitraryModeShotDistributor::Pointer shotDist  336 : 46 : = new ArbitraryModeShotDistributor( calc );  337 : 46 : distributors.push\_back( shotDist );  338 : : DivisionParams \*divParam = new DivisionParams(  339 : : mParams->getSlopeData(),  340 : 46 : mParams->getDIVTable());  341 : 46 : PatternProcessor \*patProc = new PatternProcessor();  342 : 46 : patProc->setThreadID( thread );  343 : 46 : patProc->setMode( divMode );  344 : 46 : patProc->setDivisionParams( divParam );  345 : 46 : patProc->registerDistributor( boost::ref(\*shotDist) );  346 : : ArbitraryModeCellProcessor::Pointer cellProc  347 : : = new ArbitraryModeCellProcessor(  348 : 46 : thread, mOutputGen, patProc, shotDist );  349 : 46 : processor->addCellProcessor( cellProc );  350 : 46 : cellProc->setScheduler( processor );  351 : 46 : cellProcs.push\_back( std::make\_pair(cellProc,false) );  352 : 46 : mThreadManager.startThread( cellProc );  353 : : }  354 : : //TODO::ANALYZE  355 : 46 : mThreadManager.startThread( mOutputGen );  356 : 46 : struct timeval startTime;  357 : 46 : gettimeofday( &startTime, NULL );  358 : 46 : mThreadManager.startThread( processor );  359 : : //All threads started .. now wait for them to complete  360 : 46 : mThreadManager.waitForCompletion();  361 : 46 : struct timeval endTime;  362 : 46 : gettimeofday( &endTime, NULL );  363 : :   364 : : long long usecs = ( endTime.tv\_sec - startTime.tv\_sec) \* 1000000  365 : 46 : + ( endTime.tv\_usec - startTime.tv\_usec);  366 : 46 : e9k\_info("Conversion time: %lld sec %lld usec\n",  367 : : usecs / 1000000,  368 : : usecs % 1000000  369 : : );  370 : :   371 : 46 : SDDB \*pSDDB = SDDB::instance();  372 : 46 : try  373 : : {  374 : 46 : pSDDB->conversionCompleted();  375 : : }  376 [ # # ]: 0 : catch( DPL::Error \*pChild ) {  377 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  378 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  379 : : ErrorHandler::eErrFunction,  380 : : pErr->getErrorMsg(  381 : : ErrorHandler::eErrFunction,  382 : : "SDDB::conversionCompleted"));  383 : : }  384 : :   385 : 92 : for( ArbitraryModeCellProcessorList::iterator it = cellProcs.begin();  386 : : it != cellProcs.end(); ++it )  387 : : {  388 : 46 : ArbitraryModeCellProcessor::Pointer cellProc = (\*it).first;  389 [ + - ]: 46 : delete cellProc;  390 : : }  391 : 46 : for( std::vector<ArbitraryModeShotDistributor::Pointer>::iterator it  392 : 92 : = distributors.begin(); it != distributors.end(); ++it )  393 : : {  394 : 46 : delete (\*it);  395 : : }  396 : 46 : cellProcs.clear();  397 : 46 : distributors.clear();  398 [ + - ]: 46 : delete processor;  399 [ + + ]: 46 : if( DPL::Error \*pErrObj = mThreadManager.getErrorObject() )  400 : : {  401 : 2 : ErrorHandler \*pErr = ErrorHandler::instance();  402 : 2 : pErr->errorAdd(pErrObj, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  403 : : ErrorHandler::eErrFunction,  404 : : pErr->getErrorMsg( ErrorHandler::eErrFunction,  405 : : "ThreadManager::startThread"));  406 : : }  407 : : }  408 : :   409 : : void  410 : 1094 : ConversionEngine::setupShotTimeCalc() {  411 : 1094 : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg = mParams->getStartupMessage();  412 : 1094 : const sgs\_prox\_param\_t &proxParams = mParams->getColumnParams()->prox;  413 : 1094 : const sgs\_doc\_param\_t &docParams = mParams->getColumnParams()->doc;  414 : :   415 : : ShotTimeCalculator::ProxMode proxMode  416 : : = static\_cast<ShotTimeCalculator::ProxMode>(  417 : 1094 : proxParams.mode & 0x00000001);  418 : : ShotTimeCalculator calc(  419 : : proxMode,  420 : : mParams->getDoseTable(),  421 : : mParams->getKBRTable(),  422 : 1094 : mParams->getDOCTable() );  423 : : double stdEta = proxParams.eta[0] + proxParams.eta[1]  424 : 1094 : + proxParams.eta[2] + proxParams.eta[3];  425 : :   426 : 1094 : calc.precalculate( proxMode, docParams.std\_dose,  427 : : docParams.current, stdEta);  428 : :   429 : :   430 : 1094 : mBlkUnit = 1.0 / ShotTimeCalculator::getBlkUnit();  431 : : uint16\_t MIN\_TP = static\_cast<uint16\_t>(  432 : 1094 : pStartMsg->dMinShotTime \* mBlkUnit);  433 : : uint16\_t ADJ\_TP = static\_cast<uint16\_t>(  434 : 1094 : pStartMsg->dNormShotTime \* mBlkUnit);  435 : 1094 : mShotOffset = static\_cast<uint16\_t>(  436 : : docParams.shot\_offset \* mBlkUnit);  437 : :   438 [ - + ]: 1094 : if( MIN\_TP <= mShotOffset ){  439 : 0 : MIN\_TP = 0;  440 : : }  441 : : else  442 : : {  443 : 1094 : MIN\_TP = static\_cast<uint16\_t>( ( MIN\_TP - mShotOffset )  444 : : / pStartMsg->dTimeDependDoseCorrection );  445 : 1094 : ADJ\_TP = static\_cast<uint16\_t>( ( ADJ\_TP - mShotOffset )  446 : : / pStartMsg->dTimeDependDoseCorrection );  447 : : }  448 : :   449 : 1094 : ShotTimeCalculator::setMIN\_TP( MIN\_TP );  450 : 1094 : ShotTimeCalculator::setADJ\_TP( ADJ\_TP );  451 : :   452 : : #ifdef ASTL  453 : : mShotTimeStep = (uint16\_t)( 1.6 \* mBlkUnit );  454 : : mShotTimeThr = (uint16\_t)( 0.1 \* mBlkUnit );  455 : : #else  456 : 1094 : mShotTimeStep = (uint16\_t)( 3.2 \* mBlkUnit );  457 : 1094 : mShotTimeThr = 1;  458 : : #endif   459 : : }  460 : : void  461 : : ConversionEngine::dumpConvConfParams()  462 : 1063 : {  463 : :   464 : : ////////////////////////////DUMPING FIRST PARAMS//////////////////////////////////   465 : :   466 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  467 : : "\*\*\*\*\*\*\*\*\*\*\*\*\*Dumping FIRST PARAMS\*\*\*\*\*\*\*\*\*\*\*\*\*");  468 : :   469 : : const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg  470 : 1063 : = mParams->getStartupMessage();  471 : :   472 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  473 : : "uiDpbProcNo",  474 : : pStartMsg->uiDpbProcNo);  475 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  476 : : "uiAU",  477 : : pStartMsg->uiAU);  478 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  479 : : "usProcessingMode",   480 : : pStartMsg->usProcessingMode);  481 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  482 : : "usConcurrency",   483 : : pStartMsg->usConcurrency);  484 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  485 : : "usFlag",   486 : : pStartMsg->usFlag);  487 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  488 : : "usDiag",   489 : : pStartMsg->usDiag);  490 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  491 : : "uiSlidingWindowSFGCount",   492 : : pStartMsg->uiSlidingWindowSFGCount);  493 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  494 : : "uiRetryConstantIntervalTime",   495 : : pStartMsg->uiRetryConstantIntervalTime);  496 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  497 : : "uiRetryConstantCount",   498 : : pStartMsg->uiRetryConstantCount);  499 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  500 : : "uiRetryMaxTime",   501 : : pStartMsg->uiRetryMaxTime);  502 : : /\*Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  503 : : "iDummy",   504 : : pStartMsg->iDummy);\*/  505 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  506 : : "ullSdfRemoteMemoryID",  507 : : pStartMsg->ullSdfRemoteMemoryID);  508 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  509 : : "ullRefRemoteMemoryID",   510 : : pStartMsg->ullRefRemoteMemoryID);  511 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  512 : : "ullLinkRemoteMemoryID",   513 : : pStartMsg->ullLinkRemoteMemoryID);  514 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  515 : : "ullCellRemoteMemoryID",   516 : : pStartMsg->ullCellRemoteMemoryID);  517 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  518 : : "ullCellCommonRemoteMemoryID",   519 : : pStartMsg->ullCellCommonRemoteMemoryID);  520 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  521 : : "ullDoseRemoteMemoryID",   522 : : pStartMsg->ullDoseRemoteMemoryID);  523 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  524 : : "ullSlopeRemoteMemoryID",   525 : : pStartMsg->ullSlopeRemoteMemoryID);  526 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  527 : : "ullDxpTblRemoteMemoryID",   528 : : pStartMsg->ullDxpTblRemoteMemoryID);  529 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  530 : : "ullDxpMapRemoteMemoryID",   531 : : pStartMsg->ullDxpMapRemoteMemoryID);  532 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  533 : : "ullDivRemoteMemoryID",   534 : : pStartMsg->ullDivRemoteMemoryID);  535 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  536 : : "ullDocRemoteMemoryID",   537 : : pStartMsg->ullDocRemoteMemoryID);  538 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  539 : : "ullGmcRemoteMemoryID",   540 : : pStartMsg->ullGmcRemoteMemoryID);  541 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  542 : : "ullAITblRemoteMemoryID",   543 : : pStartMsg->ullAITblRemoteMemoryID);  544 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  545 : : "ullPdgRemoteMemoryID",   546 : : pStartMsg->ullPdgRemoteMemoryID);  547 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  548 : : "ullMainSetlTblRemoteMemoryID",   549 : : pStartMsg->ullMainSetlTblRemoteMemoryID);  550 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  551 : : "ullSubSetlTblRemoteMemoryID",   552 : : pStartMsg->ullSubSetlTblRemoteMemoryID);  553 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  554 : : "ullTDSetlTblRemoteMemoryID",   555 : : pStartMsg->ullTDSetlTblRemoteMemoryID);  556 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  557 : : "ullSizerSetlTblRemoteMemoryID",   558 : : pStartMsg->ullSizerSetlTblRemoteMemoryID);  559 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  560 : : "uiDxpTblAccessMethod",   561 : : pStartMsg->uiDxpTblAccessMethod);  562 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  563 : : "uiDxpMapAccessMethod",   564 : : pStartMsg->uiDxpMapAccessMethod);  565 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  566 : : "uiDivTblAccessMethod",   567 : : pStartMsg->uiDivTblAccessMethod);  568 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  569 : : "uiDocTblAccessMethod",   570 : : pStartMsg->uiDocTblAccessMethod);  571 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  572 : : "uiGmcMapAccessMethod",   573 : : pStartMsg->uiGmcMapAccessMethod);  574 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  575 : : "uiAITblAccessMethod",   576 : : pStartMsg->uiAITblAccessMethod);  577 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  578 : : "uiPdg12xAccessMethod",   579 : : pStartMsg->uiPdg12xAccessMethod);  580 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  581 : : "uiMainSetlTblAccessMethod",   582 : : pStartMsg->uiMainSetlTblAccessMethod);  583 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  584 : : "uiSubSetlTblAccessMethod",   585 : : pStartMsg->uiSubSetlTblAccessMethod);  586 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  587 : : "uiTDSetlTblAccessMethod",   588 : : pStartMsg->uiTDSetlTblAccessMethod);  589 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  590 : : "uiSizerSetlTblAccessMethod",   591 : : pStartMsg->uiSizerSetlTblAccessMethod);  592 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  593 : : "iDxpTblSharedMemoryID",   594 : : pStartMsg->iDxpTblSharedMemoryID);  595 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  596 : : "iDxpMapSharedMemoryID",   597 : : pStartMsg->iDxpMapSharedMemoryID);  598 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  599 : : "iDivSharedMemoryID",   600 : : pStartMsg->iDivSharedMemoryID);  601 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  602 : : "iDocSharedMemoryID",   603 : : pStartMsg->iDocSharedMemoryID);  604 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  605 : : "iGmcSharedMemoryID",   606 : : pStartMsg->iGmcSharedMemoryID);  607 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  608 : : "iAITblSharedMemoryID",   609 : : pStartMsg->iAITblSharedMemoryID);  610 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  611 : : "iPdg12xSharedMemoryID",   612 : : pStartMsg->iPdg12xSharedMemoryID);  613 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  614 : : "iMainSetlTblSharedMemoryID",   615 : : pStartMsg->iMainSetlTblSharedMemoryID);  616 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  617 : : "iSubSetlTblSharedMemoryID",   618 : : pStartMsg->iSubSetlTblSharedMemoryID);  619 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  620 : : "iTDSetlTblSharedMemoryID",   621 : : pStartMsg->iTDSetlTblSharedMemoryID);  622 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  623 : : "iSizerSetlTblSharedMemoryID",   624 : : pStartMsg->iSizerSetlTblSharedMemoryID);  625 : :   626 : : /\*double llVchipOriginX =   627 : : static\_cast<double>((pStartMsg->llVchipOriginX \* pStartMsg->uiAU)/1000000000.0);   628 : : double llVchipOriginY =   629 : : static\_cast<double>((pStartMsg->llVchipOriginX \* pStartMsg->uiAU)/1000000000.0); \*/  630 : :   631 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  632 : : "llVchipOriginX",   633 : : pStartMsg->llVchipOriginX);  634 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  635 : : "llVchipOriginY",   636 : : pStartMsg->llVchipOriginY);  637 : : /\*  638 : : double ullVchipSizeX =   639 : : static\_cast<double>((pStartMsg->ullVchipSizeX \* pStartMsg->uiAU)/1000000000.0);   640 : : double ullVchipSizeY =   641 : : static\_cast<double>((pStartMsg->ullVchipSizeY \* pStartMsg->uiAU)/1000000000.0);   642 : : \*/  643 : : //Not presently present in converter.conf  644 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  645 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  646 : : "ullVchipSizeX",   647 : : pStartMsg->ullVchipSizeX);  648 : : // ullVchipSizeX);  649 : : //Not presently present in converter.conf  650 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  651 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  652 : : "ullVchipSizeY",   653 : : pStartMsg->ullVchipSizeY);  654 : : // ullVchipSizeY);  655 : : //Not presently present in converter.conf  656 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  657 : : "ullVchipBSizeX",   658 : : pStartMsg->ullVchipBSizeX);  659 : : //Not presently present in converter.conf  660 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  661 : : "ullVchipBSizeY",   662 : : pStartMsg->ullVchipBSizeY);  663 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  664 : : "usFrameOrientation",   665 : : pStartMsg->usFrameOrientation);  666 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  667 : : "sDummy1",   668 : : pStartMsg->sDummy1);  669 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  670 : : "uiAvgShotCountInTF",   671 : : pStartMsg->uiAvgShotCountInTF);  672 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  673 : : "ullCellMaxSizeX",   674 : : pStartMsg->ullCellMaxSizeX);  675 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  676 : : "ullCellMaxSizeY",   677 : : pStartMsg->ullCellMaxSizeY);  678 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  679 : : "ullArrCellMaxSizeX",   680 : : pStartMsg->ullArrCellMaxSizeX);  681 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  682 : : "ullArrCellMaxSizeY",   683 : : pStartMsg->ullArrCellMaxSizeY);  684 : : /\*  685 : : double llDpbX =   686 : : static\_cast<double>((pStartMsg->llDpbX \* pStartMsg->uiAU)/1000000000.0);   687 : :   688 : : double llDpbY =   689 : : static\_cast<double>((pStartMsg->llDpbY \* pStartMsg->uiAU)/1000000000.0);   690 : : \*/  691 : : //Not presently present in converter.conf  692 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  693 : : "llDpbX",   694 : : pStartMsg->llDpbX);  695 : : //llDpbX);  696 : : //Not presently present in converter.conf  697 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  698 : : "llDpbY",   699 : : pStartMsg->llDpbY);  700 : : // llDpbY);  701 : : /\*  702 : : double ullDpbWidth =   703 : : static\_cast<double>((pStartMsg->ullDpbWidth \* pStartMsg->uiAU)/1000000000.0);   704 : :   705 : : double ullDpbHeight =   706 : : static\_cast<double>((pStartMsg->ullDpbHeight \* pStartMsg->uiAU)/1000000000.0);   707 : : \*/  708 : :   709 : : //Not presently present in converter.conf  710 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  711 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  712 : : "ullDpbWidth",  713 : : pStartMsg->ullDpbWidth);   714 : : //ullDpbWidth);  715 : : //Not presently present in converter.conf  716 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  717 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  718 : : "ullDpbHeight",  719 : : pStartMsg->ullDpbHeight);   720 : : //ullDpbHeight);  721 : : /\*  722 : : double llPtnProcessOriginX =   723 : : static\_cast<double>((pStartMsg->llPtnProcessOriginX  724 : : \* pStartMsg->uiAU)/1000000000.0);   725 : :   726 : : double llPtnProcessOriginY =   727 : : static\_cast<double>((pStartMsg->llPtnProcessOriginY  728 : : \* pStartMsg->uiAU)/1000000000.0);   729 : : \*/  730 : :   731 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  732 : : "llPtnProcessOriginX",   733 : : pStartMsg->llPtnProcessOriginX);  734 : : //llPtnProcessOriginX);  735 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  736 : : "llPtnProcessOriginY",  737 : : pStartMsg->llPtnProcessOriginY);   738 : : //llPtnProcessOriginY);  739 : : /\*  740 : : double ullPtnProcessWidth =   741 : : static\_cast<double>((pStartMsg->ullPtnProcessWidth  742 : : \* pStartMsg->uiAU)/1000000000.0);   743 : :   744 : : double ullPtnProcessHeight =   745 : : static\_cast<double>((pStartMsg->ullPtnProcessHeight  746 : : \* pStartMsg->uiAU)/1000000000.0);   747 : :   748 : : \*/  749 : :   750 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  751 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  752 : : "ullPtnProcessWidth",   753 : : pStartMsg->ullPtnProcessWidth);  754 : : //ullPtnProcessWidth);  755 : : //Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  756 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  757 : : "ullPtnProcessHeight",   758 : : pStartMsg->ullPtnProcessHeight);  759 : : //ullPtnProcessHeight);  760 : :   761 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  762 : : "uiSTLCountInColumn",   763 : : pStartMsg->uiSTLCountInColumn);  764 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  765 : : "uiSTLNo",   766 : : pStartMsg->uiSTLNo);  767 : :   768 : : //Not presently present in converter.conf  769 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  770 : : "uiMaxSFLID",   771 : : pStartMsg->uiMaxSFLID);  772 : :   773 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  774 : : "uiSFLCountInColumn",   775 : : pStartMsg->uiSFLCountInColumn);  776 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  777 : : "uiSFLCountInDpb",   778 : : pStartMsg->uiSFLCountInDpb);  779 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  780 : : "uiRepCount",   781 : : pStartMsg->uiRepCount);  782 : : /\*   783 : : double iSFSizeX =   784 : : static\_cast<double>((pStartMsg->iSFSizeX  785 : : \* pStartMsg->uiAU)/1000000000.0);   786 : : double iSFSizeY =   787 : : static\_cast<double>((pStartMsg->iSFSizeY  788 : : \* pStartMsg->uiAU)/1000000000.0);   789 : : \*/  790 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  791 : : "iSFSizeX",   792 : : pStartMsg->iSFSizeX);  793 : : //iSFSizeX);  794 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  795 : : "iSFSizeY",  796 : : pStartMsg->iSFSizeY);   797 : : //iSFSizeY);  798 : :   799 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  800 : : "usSFOrder",   801 : : pStartMsg->usSFOrder);  802 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  803 : : "usTFPriority",   804 : : pStartMsg->usTFPriority);  805 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  806 : : "usTFLocationOrder",   807 : : pStartMsg->usTFLocationOrder);  808 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  809 : : "usTFShapeOrder",   810 : : pStartMsg->usTFShapeOrder);  811 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  812 : : "usSegmentCountX",   813 : : pStartMsg->usSegmentCountX);  814 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  815 : : "usSegmentCountY",   816 : : pStartMsg->usSegmentCountY);  817 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  818 : : "usMaxShotShiftCount",   819 : : pStartMsg->usMaxShotShiftCount);  820 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  821 : : "sDummy2",   822 : : pStartMsg->sDummy2);  823 : : /\*  824 : : double iTFSizeX =   825 : : static\_cast<double>((pStartMsg->iTFSizeX  826 : : \* pStartMsg->uiAU)/1000000000.0);   827 : : double iTFSizeY =   828 : : static\_cast<double>((pStartMsg->iTFSizeY  829 : : \* pStartMsg->uiAU)/1000000000.0);   830 : : \*/  831 : :   832 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  833 : : "iTFSizeX",   834 : : pStartMsg->iTFSizeX);  835 : : //iTFSizeX);  836 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  837 : : "iTFSizeY",  838 : : pStartMsg->iTFSizeY);   839 : : //iTFSizeY);  840 : : /\*  841 : : double iSwingBackOffsetX1 =   842 : : static\_cast<double>((pStartMsg->iSwingBackOffsetX1  843 : : \* pStartMsg->uiAU)/1000000000.0);   844 : : double iSwingBackOffsetY1 =   845 : : static\_cast<double>((pStartMsg->iSwingBackOffsetY1  846 : : \* pStartMsg->uiAU)/1000000000.0);   847 : : \*/  848 : :   849 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  850 : : "iSwingBackOffsetX1",   851 : : pStartMsg->iSwingBackOffsetX1);  852 : : //iSwingBackOffsetX1);  853 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  854 : : "iSwingBackOffsetY1",  855 : : pStartMsg->iSwingBackOffsetY1);   856 : : //iSwingBackOffsetY1);  857 : : /\*  858 : : double iSwingBackOffsetX2 =   859 : : static\_cast<double>((pStartMsg->iSwingBackOffsetX2  860 : : \* pStartMsg->uiAU)/1000000000.0);   861 : : double iSwingBackOffsetY2 =   862 : : static\_cast<double>((pStartMsg->iSwingBackOffsetY2  863 : : \* pStartMsg->uiAU)/1000000000.0);   864 : : \*/  865 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  866 : : "iSwingBackOffsetX2",  867 : : pStartMsg->iSwingBackOffsetX2);   868 : : //iSwingBackOffsetX2);  869 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  870 : : "iSwingBackOffsetY2",  871 : : pStartMsg->iSwingBackOffsetY2);  872 : : //iSwingBackOffsetY2);  873 : : /\*  874 : : double iSwingBackOffsetX3 =   875 : : static\_cast<double>((pStartMsg->iSwingBackOffsetX3  876 : : \* pStartMsg->uiAU)/1000000000.0);   877 : : double iSwingBackOffsetY3 =   878 : : static\_cast<double>((pStartMsg->iSwingBackOffsetY3  879 : : \* pStartMsg->uiAU)/1000000000.0);   880 : : \*/  881 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  882 : : "iSwingBackOffsetX3",  883 : : pStartMsg->iSwingBackOffsetX3);   884 : : //iSwingBackOffsetX3);  885 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  886 : : "iSwingBackOffsetY3",  887 : : pStartMsg->iSwingBackOffsetY3);  888 : : //iSwingBackOffsetY3);  889 : : /\*  890 : : double iSwingBackOffsetX4 =   891 : : static\_cast<double>((pStartMsg->iSwingBackOffsetX4  892 : : \* pStartMsg->uiAU)/1000000000.0);   893 : : double iSwingBackOffsetY4 =   894 : : static\_cast<double>((pStartMsg->iSwingBackOffsetY4  895 : : \* pStartMsg->uiAU)/1000000000.0);  896 : : \*/   897 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  898 : : "iSwingBackOffsetX4",  899 : : pStartMsg->iSwingBackOffsetX4);  900 : : //iSwingBackOffsetX4);  901 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  902 : : "iSwingBackOffsetY4",  903 : : pStartMsg->iSwingBackOffsetY4);   904 : : //iSwingBackOffsetY4);  905 : : /\*  906 : : double iSwingBackOffsetX5 =   907 : : static\_cast<double>((pStartMsg->iSwingBackOffsetX5  908 : : \* pStartMsg->uiAU)/1000000000.0);   909 : : double iSwingBackOffsetY5 =   910 : : static\_cast<double>((pStartMsg->iSwingBackOffsetY5  911 : : \* pStartMsg->uiAU)/1000000000.0);   912 : : \*/  913 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  914 : : "iSwingBackOffsetX5",  915 : : pStartMsg->iSwingBackOffsetX5);   916 : : //iSwingBackOffsetX5);  917 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  918 : : "iSwingBackOffsetY5",  919 : : pStartMsg->iSwingBackOffsetY5);   920 : : //iSwingBackOffsetY5);  921 : :   922 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  923 : : "dShotTimeResolution",  924 : : pStartMsg->dShotTimeResolution);   925 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  926 : : "dSettlingTimeResolution",  927 : : pStartMsg->dSettlingTimeResolution);   928 : : /\*Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  929 : : "dTFSettlingTime",   930 : : pStartMsg->dTFSettlingTime);\*/  931 : :   932 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  933 : : "uiSFStartNullShotNum",   934 : : pStartMsg->uiSFStartNullShotNum);  935 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  936 : : "uiSFEndNullShotNum",   937 : : pStartMsg->uiSFEndNullShotNum);  938 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  939 : : "dSFStartNullShotTime",   940 : : pStartMsg->dSFStartNullShotTime);  941 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  942 : : "dSFEndNullShotTime",   943 : : pStartMsg->dSFEndNullShotTime);  944 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  945 : : "dSFStartNullSettlingTime",   946 : : pStartMsg->dSFStartNullSettlingTime);  947 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  948 : : "dSFEndNullSettlingTime",   949 : : pStartMsg->dSFEndNullSettlingTime);  950 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  951 : : "uiDrawTimeCalcMode",   952 : : pStartMsg->uiDrawTimeCalcMode);  953 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  954 : : "uiNumRowDrawTimeCalc",   955 : : pStartMsg->uiNumRowDrawTimeCalc);  956 : : /\*   957 : : double llDrawTimeCalcOriginX =   958 : : static\_cast<double>((pStartMsg->llDrawTimeCalcOriginX  959 : : \* pStartMsg->uiAU)/1000000000.0);   960 : : double llDrawTimeCalcPitch =   961 : : static\_cast<double>((pStartMsg->llDrawTimeCalcPitch  962 : : \* pStartMsg->uiAU)/1000000000.0);   963 : : \*/  964 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  965 : : "llDrawTimeCalcOriginX",  966 : : pStartMsg->llDrawTimeCalcOriginX);   967 : : //llDrawTimeCalcOriginX);  968 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  969 : : "llDrawTimeCalcPitch",   970 : : pStartMsg->llDrawTimeCalcPitch);  971 : : //llDrawTimeCalcPitch);  972 : :   973 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  974 : : "llMainSettlingOffset",   975 : : pStartMsg->llMainSettlingOffset);  976 : : /\*Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  977 : : "llSGSoffs",   978 : : pStartMsg->llSGSoffs);\*/  979 : :   980 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  981 : : "llMinSFTime",   982 : : pStartMsg->llMinSFTime);  983 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  984 : : "dMinShotTime",   985 : : pStartMsg->dMinShotTime);  986 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  987 : : "dNormShotTime",   988 : : pStartMsg->dNormShotTime);  989 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  990 : : "dTimeDependDoseCorrection",   991 : : pStartMsg->dTimeDependDoseCorrection);  992 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  993 : : "dShotTimeGain",   994 : : pStartMsg->dShotTimeGain);  995 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  996 : : "dShotCycleTimeScaleFactor",   997 : : pStartMsg->dShotCycleTimeScaleFactor);  998 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  999 : : "ullEstimatedShotCount",   1000 : : pStartMsg->ullEstimatedShotCount);  1001 : :   1002 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  1003 : : "\*\*\*\*\*\*\*\*\*\*End Dumping FIRST PARAMS\*\*\*\*\*\*\*\*\*\*\*");  1004 : : ////////////////////////END DUMPING FIRST PARAMS///////////////////////  1005 : :   1006 : : //////////////////////// DUMPING SFL INFO/////////////////////////  1007 : : const DPMANAGER\_MESSAGE\_SFL\_INFO \*inSFLInfo =   1008 : 1063 : ConverterParams::instance()->getLayerParams();  1009 [ + - + + ]: 2195 : for(uint32\_t i = 0; i < pStartMsg->uiSFLCountInDpb ; i++)  1010 : : {  1011 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  1012 : : "\*\*\*\*\*\*\*\*\*\*Dumping SFL Info\*\*\*\*\*\*\*\*\*\*\*");  1013 : 1132 : const DPMANAGER\_MESSAGE\_SFL\_INFO \*pLayer = inSFLInfo + i;  1014 : :   1015 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1016 : : "uiSFLNo",  1017 : : pLayer->uiSFLNo);  1018 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1019 : : "uiSFLID",  1020 : : pLayer->uiSFLID);  1021 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1022 : : "llSFLOriginX",  1023 : : pLayer->llSFLOriginX);  1024 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1025 : : "llSFLOriginY",  1026 : : pLayer->llSFLOriginY);  1027 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1028 : : "uiSFCountX",  1029 : : pLayer->uiSFCountX);  1030 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1031 : : "uiSFCountY",  1032 : : pLayer->uiSFCountY);  1033 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1034 : : "llTFOriginX",  1035 : : pLayer->llTFOriginX);  1036 : :   1037 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1038 : : "llTFOriginY",  1039 : : pLayer->llTFOriginY);  1040 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1041 : : "usTFCountX",  1042 : : pLayer->usTFCountX);  1043 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1044 : : "usTFCountY",  1045 : : pLayer->usTFCountY);  1046 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1047 : : "iDummy",  1048 : : pLayer->iDummy);  1049 : 1132 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  1050 : : "\*\*\*\*\*\*\*\*\*\*End Dumping SFL Info\*\*\*\*\*\*\*\*\*\*\*");  1051 : : }  1052 : :   1053 : :   1054 : : ////////////////////////END DUMPING SFL INFO///////////////////////  1055 : :   1056 : : //////////////////////// DUMPING SECOND PARAMS/////////////////////////  1057 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  1058 : : "\*\*\*\*\*\*\*\*\*\*Dumping SECOND PARAMS\*\*\*\*\*\*\*\*\*\*\*");  1059 : : const sgs\_column\_t \*inSecParam =   1060 : 1063 : ConverterParams::instance()->getColumnParams();  1061 : :   1062 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1063 : : "unit.laser\_wavelength",  1064 : : inSecParam->unit.laser\_wavelength);  1065 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1066 : : "unit.laser\_count",  1067 : : inSecParam->unit.laser\_count);  1068 : :   1069 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1070 : : "mask.mask\_center\_x",  1071 : : inSecParam->mask.mask\_center\_x);  1072 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1073 : : "mask.mask\_center\_y",  1074 : : inSecParam->mask.mask\_center\_y);  1075 : :   1076 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_STRING\_PARAMS,  1077 : : "clmn\_inf.chip\_name",  1078 : : inSecParam->clmn\_inf.chip\_name);  1079 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1080 : : "clmn\_inf.cp\_org\_st\_x",  1081 : : inSecParam->clmn\_inf.cp\_org\_st\_x);  1082 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1083 : : "clmn\_inf.cp\_org\_st\_y",  1084 : : inSecParam->clmn\_inf.cp\_org\_st\_y);  1085 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1086 : : "clmn\_inf.format",  1087 : : inSecParam->clmn\_inf.format);  1088 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1089 : : "clmn\_inf.chip\_size\_x",  1090 : : inSecParam->clmn\_inf.chip\_size\_x);  1091 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1092 : : "clmn\_inf.chip\_size\_y",  1093 : : inSecParam->clmn\_inf.chip\_size\_y);  1094 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1095 : : "clmn\_inf.w\_stripe",  1096 : : inSecParam->clmn\_inf.w\_stripe);  1097 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1098 : : "clmn\_inf.spd\_cntl",  1099 : : inSecParam->clmn\_inf.spd\_cntl);  1100 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1101 : : "clmn\_inf.fwd\_flg",  1102 : : inSecParam->clmn\_inf.fwd\_flg);  1103 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1104 : : "clmn\_inf.w\_crst\_siz",  1105 : : inSecParam->clmn\_inf.w\_crst\_siz);  1106 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1107 : : "clmn\_inf.h\_crst\_siz",  1108 : : inSecParam->clmn\_inf.h\_crst\_siz);  1109 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1110 : : "clmn\_inf.w\_sub\_fld",  1111 : : inSecParam->clmn\_inf.w\_sub\_fld);  1112 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1113 : : "clmn\_inf.h\_sub\_fld",  1114 : : inSecParam->clmn\_inf.h\_sub\_fld);  1115 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1116 : : "clmn\_inf.sub\_fld\_mode",  1117 : : inSecParam->clmn\_inf.sub\_fld\_mode);  1118 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1119 : : "clmn\_inf.pile\_time",  1120 : : inSecParam->clmn\_inf.pile\_time);  1121 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1122 : : "clmn\_inf.stripe\_pile",  1123 : : inSecParam->clmn\_inf.stripe\_pile);  1124 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1125 : : "clmn\_inf.sub\_fld\_pile",  1126 : : inSecParam->clmn\_inf.sub\_fld\_pile);  1127 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_uLONG\_PARAMS,  1128 : : "clmn\_inf.w\_stripe\_sft",  1129 : : inSecParam->clmn\_inf.w\_stripe\_sft);  1130 : : //int64  1131 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1132 : : "clmn\_inf.dpb\_mode",  1133 : : inSecParam->clmn\_inf.dpb\_mode);  1134 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1135 : : "clmn\_inf.dpb\_size\_x",  1136 : : inSecParam->clmn\_inf.dpb\_size\_x);  1137 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1138 : : "clmn\_inf.sf\_range\_x",  1139 : : inSecParam->clmn\_inf.sf\_range\_x);  1140 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1141 : : "clmn\_inf.sf\_range\_y",  1142 : : inSecParam->clmn\_inf.sf\_range\_y);  1143 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1144 : : "clmn\_inf.sf\_fringe\_x",  1145 : : inSecParam->clmn\_inf.sf\_fringe\_x);  1146 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1147 : : "clmn\_inf.sf\_fringe\_y",  1148 : : inSecParam->clmn\_inf.sf\_fringe\_y);  1149 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1150 : : "clmn\_inf.multi\_writing\_count",  1151 : : inSecParam->clmn\_inf.multi\_writing\_count);  1152 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1153 : : "clmn\_inf.multi\_writing\_mode",  1154 : : inSecParam->clmn\_inf.multi\_writing\_mode);  1155 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1156 : : "clmn\_inf.st\_layer\_count",  1157 : : inSecParam->clmn\_inf.st\_layer\_count);  1158 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1159 : : "clmn\_inf.sf\_layer\_count",  1160 : : inSecParam->clmn\_inf.sf\_layer\_count);  1161 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1162 : : "clmn\_inf.sfg\_repetition",  1163 : : inSecParam->clmn\_inf.sfg\_repetition);  1164 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1165 : : "clmn\_inf.st\_shift",  1166 : : inSecParam->clmn\_inf.st\_shift);  1167 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1168 : : "clmn\_inf.sf\_shift",  1169 : : inSecParam->clmn\_inf.sf\_shift);  1170 : : //double shot\_volume\_thr  1171 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1172 : : "clmn\_inf.shot\_volume\_thr",  1173 : : inSecParam->clmn\_inf.shot\_volume\_thr);  1174 : : //int64\_t clmn\_base\_grid  1175 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_LONG\_PARAMS,  1176 : : "clmn\_inf.clmn\_base\_grid",  1177 : : inSecParam->clmn\_inf.clmn\_base\_grid);  1178 : : //bool dose\_ratio\_skip  1179 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1180 : : "clmn\_inf.dose\_ratio\_skip",  1181 : : inSecParam->clmn\_inf.dose\_ratio\_skip);  1182 : : //int32\_t stsfsw  1183 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_INT32\_PARAMS,  1184 : : "clmn\_inf.stsfsw",  1185 : : inSecParam->clmn\_inf.stsfsw);  1186 : :   1187 : : //TODO: StSf\_ParamInfo   1188 : : /\*Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1189 : : "clmn\_inf.",  1190 : : inSecParam->clmn\_inf.);  1191 : : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1192 : : "clmn\_inf.",  1193 : : inSecParam->clmn\_inf.);\*/  1194 : : ///Dumping PROX parameters   1195 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1196 : : "prox.mode",  1197 : : inSecParam->prox.mode);  1198 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1199 : : "prox.kernels",  1200 : : inSecParam->prox.kernels);  1201 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1202 : : "prox.mesh\_size",  1203 : : inSecParam->prox.mesh\_size);  1204 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_arrFLOAT\_PARAMS,  1205 : : "prox.eta",  1206 : : inSecParam->prox.eta[0],  1207 : : inSecParam->prox.eta[1],  1208 : : inSecParam->prox.eta[2],  1209 : : inSecParam->prox.eta[3]);  1210 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_arrFLOAT\_PARAMS,  1211 : : "prox.sigma",  1212 : : inSecParam->prox.sigma[0],  1213 : : inSecParam->prox.sigma[1],  1214 : : inSecParam->prox.sigma[2],  1215 : : inSecParam->prox.sigma[3]);  1216 : :   1217 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1218 : : "prox.e\_map.mesh\_size\_x",  1219 : : inSecParam->prox.e\_map.mesh\_size\_x);  1220 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1221 : : "prox.e\_map.mesh\_size\_y",  1222 : : inSecParam->prox.e\_map.mesh\_size\_y);  1223 : :   1224 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1225 : : "prox.e\_map.mesh\_x",  1226 : : inSecParam->prox.e\_map.mesh\_x);  1227 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1228 : : "prox.e\_map.mesh\_y",  1229 : : inSecParam->prox.e\_map.mesh\_y);  1230 : :   1231 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1232 : : "prox.e\_map.pos\_x",  1233 : : inSecParam->prox.e\_map.pos\_x);  1234 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1235 : : "prox.e\_map.pos\_y",  1236 : : inSecParam->prox.e\_map.pos\_y);  1237 : :   1238 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1239 : : "prox.e\_map.data\_byte",  1240 : : inSecParam->prox.e\_map.data\_byte);  1241 : :   1242 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1243 : : "prox.d\_map.mesh\_size\_x",  1244 : : inSecParam->prox.d\_map.mesh\_size\_x);  1245 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1246 : : "prox.d\_map.mesh\_size\_y",  1247 : : inSecParam->prox.d\_map.mesh\_size\_y);  1248 : :   1249 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1250 : : "prox.d\_map.mesh\_x",  1251 : : inSecParam->prox.d\_map.mesh\_x);  1252 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1253 : : "prox.d\_map.mesh\_y",  1254 : : inSecParam->prox.d\_map.mesh\_y);  1255 : :   1256 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1257 : : "prox.d\_map.pos\_x",  1258 : : inSecParam->prox.d\_map.pos\_x);  1259 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1260 : : "prox.d\_map.pos\_y",  1261 : : inSecParam->prox.d\_map.pos\_y);  1262 : :   1263 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1264 : : "prox.d\_map.data\_byte",  1265 : : inSecParam->prox.d\_map.data\_byte);  1266 : :   1267 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1268 : : "prox.gradient\_no",  1269 : : inSecParam->prox.gradient\_no);  1270 : :   1271 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1272 : : "prox.dose.mesh\_size\_x",  1273 : : inSecParam->prox.dose.mesh\_size\_x);  1274 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1275 : : "prox.dose.mesh\_size\_y",  1276 : : inSecParam->prox.dose.mesh\_size\_y);  1277 : :   1278 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1279 : : "prox.dose.mesh\_x",  1280 : : inSecParam->prox.dose.mesh\_x);  1281 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1282 : : "prox.dose.mesh\_y",  1283 : : inSecParam->prox.dose.mesh\_y);  1284 : :   1285 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1286 : : "prox.dose.pos\_x",  1287 : : inSecParam->prox.dose.pos\_x);  1288 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1289 : : "prox.dose.pos\_y",  1290 : : inSecParam->prox.dose.pos\_y);  1291 : :   1292 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1293 : : "prox.dose.data\_byte",  1294 : : inSecParam->prox.dose.data\_byte);  1295 : :   1296 : :   1297 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1298 : : "prox.gradient.mesh\_size\_x",  1299 : : inSecParam->prox.gradient.mesh\_size\_x);  1300 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1301 : : "prox.gradient.mesh\_size\_y",  1302 : : inSecParam->prox.gradient.mesh\_size\_y);  1303 : :   1304 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1305 : : "prox.gradient.mesh\_x",  1306 : : inSecParam->prox.gradient.mesh\_x);  1307 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1308 : : "prox.gradient.mesh\_y",  1309 : : inSecParam->prox.gradient.mesh\_y);  1310 : :   1311 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1312 : : "prox.gradient.pos\_x",  1313 : : inSecParam->prox.gradient.pos\_x);  1314 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1315 : : "prox.gradient.pos\_y",  1316 : : inSecParam->prox.gradient.pos\_y);  1317 : :   1318 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1319 : : "prox.gradient.data\_byte",  1320 : : inSecParam->prox.gradient.data\_byte);  1321 : : ///End Dumping PROX parameters   1322 : :   1323 : : ///Dumping KBR parameters   1324 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1325 : : "kbr.mode",  1326 : : inSecParam->kbr.mode);  1327 : :   1328 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1329 : : "kbr.map.mesh\_size\_x",  1330 : : inSecParam->kbr.map.mesh\_size\_x);  1331 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1332 : : "kbr.map.mesh\_size\_y",  1333 : : inSecParam->kbr.map.mesh\_size\_y);  1334 : :   1335 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1336 : : "kbr.map.mesh\_x",  1337 : : inSecParam->kbr.map.mesh\_x);  1338 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1339 : : "kbr.map.mesh\_y",  1340 : : inSecParam->kbr.map.mesh\_y);  1341 : :   1342 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1343 : : "kbr.map.pos\_x",  1344 : : inSecParam->kbr.map.pos\_x);  1345 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1346 : : "kbr.map.pos\_y",  1347 : : inSecParam->kbr.map.pos\_y);  1348 : :   1349 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1350 : : "kbr.map.data\_byte",  1351 : : inSecParam->kbr.map.data\_byte);  1352 : : ///End Dumping KBR parameters   1353 : :   1354 : : ///Dumping DIV parameters   1355 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1356 : : "div.mode",  1357 : : inSecParam->div.mode);  1358 : : //TODO: Dump DIV\_TH table  1359 : :   1360 : : ///End Dumping DIV parameters   1361 : :   1362 : : ///Dumping DOC parameters   1363 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1364 : : "doc.std\_dose",  1365 : : inSecParam->doc.std\_dose);  1366 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1367 : : "doc.current",  1368 : : inSecParam->doc.current);  1369 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1370 : : "doc.shot\_offset",  1371 : : inSecParam->doc.shot\_offset);  1372 : : ///End Dumping DOC parameters   1373 : :   1374 : : ///Dumping GMC parameters   1375 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1376 : : "gmc.mode",  1377 : : inSecParam->gmc.mode);  1378 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1379 : : "gmc.poly\_order",  1380 : : inSecParam->gmc.poly\_order);  1381 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_arrFLOATgmc\_PARAMS,  1382 : : "gmc.poly\_x",  1383 : : inSecParam->gmc.poly\_x[0],  1384 : : inSecParam->gmc.poly\_x[1],  1385 : : inSecParam->gmc.poly\_x[2],  1386 : : inSecParam->gmc.poly\_x[3],  1387 : : inSecParam->gmc.poly\_x[4],  1388 : : inSecParam->gmc.poly\_x[5],  1389 : : inSecParam->gmc.poly\_x[6],  1390 : : inSecParam->gmc.poly\_x[7],  1391 : : inSecParam->gmc.poly\_x[8],  1392 : : inSecParam->gmc.poly\_x[9],  1393 : : inSecParam->gmc.poly\_x[10],  1394 : : inSecParam->gmc.poly\_x[11],  1395 : : inSecParam->gmc.poly\_x[12],  1396 : : inSecParam->gmc.poly\_x[13],  1397 : : inSecParam->gmc.poly\_x[14]);  1398 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_arrFLOATgmc\_PARAMS,  1399 : : "gmc.poly\_y",  1400 : : inSecParam->gmc.poly\_y[0],  1401 : : inSecParam->gmc.poly\_y[1],  1402 : : inSecParam->gmc.poly\_y[2],  1403 : : inSecParam->gmc.poly\_y[3],  1404 : : inSecParam->gmc.poly\_y[4],  1405 : : inSecParam->gmc.poly\_y[5],  1406 : : inSecParam->gmc.poly\_y[6],  1407 : : inSecParam->gmc.poly\_y[7],  1408 : : inSecParam->gmc.poly\_y[8],  1409 : : inSecParam->gmc.poly\_y[9],  1410 : : inSecParam->gmc.poly\_y[10],  1411 : : inSecParam->gmc.poly\_y[11],  1412 : : inSecParam->gmc.poly\_y[12],  1413 : : inSecParam->gmc.poly\_y[13],  1414 : : inSecParam->gmc.poly\_y[14]);  1415 : :   1416 : :   1417 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1418 : : "gmc.max\_allowable\_value",  1419 : : inSecParam->gmc.max\_allowable\_value);  1420 : :   1421 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1422 : : "gmc.map.mesh\_size\_x",  1423 : : inSecParam->gmc.map.mesh\_size\_x);  1424 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1425 : : "gmc.map.mesh\_size\_y",  1426 : : inSecParam->gmc.map.mesh\_size\_y);  1427 : :   1428 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1429 : : "gmc.map.mesh\_x",  1430 : : inSecParam->gmc.map.mesh\_x);  1431 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1432 : : "gmc.map.mesh\_y",  1433 : : inSecParam->gmc.map.mesh\_y);  1434 : :   1435 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1436 : : "gmc.map.pos\_x",  1437 : : inSecParam->gmc.map.pos\_x);  1438 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1439 : : "gmc.map.pos\_y",  1440 : : inSecParam->gmc.map.pos\_y);  1441 : :   1442 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1443 : : "gmc.map.data\_byte",  1444 : : inSecParam->gmc.map.data\_byte);  1445 : : ///End Dumping GMC parameters   1446 : :   1447 : : ///Dumping SETL TABLE:  1448 : 1063 : const SETL\_TBL &setl\_tbl = inSecParam->def.setl\_tbl;  1449 : :   1450 : : ///Dump MAIN\_SETL :   1451 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1452 : : "setl\_tbl.main.setl\_main\_first",  1453 : : setl\_tbl.main.setl\_main\_first);  1454 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1455 : : "setl\_tbl.main.setl\_main\_gain",  1456 : : setl\_tbl.main.setl\_main\_gain);  1457 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1458 : : "setl\_tbl.main.setl\_main\_offset",  1459 : : setl\_tbl.main.setl\_main\_offset);  1460 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1461 : : "setl\_tbl.main.setl\_main\_max",  1462 : : setl\_tbl.main.setl\_main\_max);  1463 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1464 : : "setl\_tbl.main.setl\_main\_min",  1465 : : setl\_tbl.main.setl\_main\_min);  1466 : :   1467 : :   1468 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1469 : : "setl\_tbl.main.setl\_main\_table.mesh\_size\_x",  1470 : : setl\_tbl.main.setl\_main\_table.mesh\_size\_x);  1471 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1472 : : "setl\_tbl.main.setl\_main\_table.mesh\_size\_y",  1473 : : setl\_tbl.main.setl\_main\_table.mesh\_size\_y);  1474 : :   1475 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1476 : : "setl\_tbl.main.setl\_main\_table.mesh\_x",  1477 : : setl\_tbl.main.setl\_main\_table.mesh\_x);  1478 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1479 : : "setl\_tbl.main.setl\_main\_table.mesh\_y",  1480 : : setl\_tbl.main.setl\_main\_table.mesh\_y);  1481 : :   1482 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1483 : : "setl\_tbl.main.setl\_main\_table.pos\_x",  1484 : : setl\_tbl.main.setl\_main\_table.pos\_x);  1485 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1486 : : "setl\_tbl.main.setl\_main\_table.pos\_y",  1487 : : setl\_tbl.main.setl\_main\_table.pos\_y);  1488 : :   1489 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1490 : : "setl\_tbl.main.setl\_main\_table.data\_byte",  1491 : : setl\_tbl.main.setl\_main\_table.data\_byte);  1492 : : ///End Dumping MAIN\_SETL  1493 : :   1494 : : ///Dump SUB\_SETL  1495 : :   1496 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1497 : : "setl\_tbl.sub.setl\_sub\_first",  1498 : : setl\_tbl.sub.setl\_sub\_first);  1499 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1500 : : "setl\_tbl.sub.setl\_sub\_gain",  1501 : : setl\_tbl.sub.setl\_sub\_gain);  1502 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1503 : : "setl\_tbl.sub.setl\_sub\_offset",  1504 : : setl\_tbl.sub.setl\_sub\_offset);  1505 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1506 : : "setl\_tbl.sub.setl\_sub\_max",  1507 : : setl\_tbl.sub.setl\_sub\_max);  1508 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1509 : : "setl\_tbl.sub.setl\_sub\_min",  1510 : : setl\_tbl.sub.setl\_sub\_min);  1511 : :   1512 : :   1513 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1514 : : "setl\_tbl.sub.setl\_sub\_table.mesh\_size\_x",  1515 : : setl\_tbl.sub.setl\_sub\_table.mesh\_size\_x);  1516 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1517 : : "setl\_tbl.sub.setl\_sub\_table.mesh\_size\_y",  1518 : : setl\_tbl.sub.setl\_sub\_table.mesh\_size\_y);  1519 : :   1520 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1521 : : "setl\_tbl.sub.setl\_sub\_table.mesh\_x",  1522 : : setl\_tbl.sub.setl\_sub\_table.mesh\_x);  1523 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1524 : : "setl\_tbl.sub.setl\_sub\_table.mesh\_y",  1525 : : setl\_tbl.sub.setl\_sub\_table.mesh\_y);  1526 : :   1527 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1528 : : "setl\_tbl.sub.setl\_sub\_table.pos\_x",  1529 : : setl\_tbl.sub.setl\_sub\_table.pos\_x);  1530 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1531 : : "setl\_tbl.sub.setl\_sub\_table.pos\_y",  1532 : : setl\_tbl.sub.setl\_sub\_table.pos\_y);  1533 : :   1534 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1535 : : "setl\_tbl.sub.setl\_sub\_table.data\_byte",  1536 : : setl\_tbl.sub.setl\_sub\_table.data\_byte);  1537 : :   1538 : : ///End Dumping SUB\_SETL  1539 : :   1540 : : //Dumping selector:  1541 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_arrFLOATselector\_PARAMS,  1542 : : "setl\_tbl.selector",  1543 : : setl\_tbl.selector[0][0],  1544 : : setl\_tbl.selector[1][0],  1545 : : setl\_tbl.selector[2][0],  1546 : : setl\_tbl.selector[3][0],  1547 : : setl\_tbl.selector[4][0],  1548 : : setl\_tbl.selector[0][1],  1549 : : setl\_tbl.selector[1][1],  1550 : : setl\_tbl.selector[2][1],  1551 : : setl\_tbl.selector[3][1],  1552 : : setl\_tbl.selector[4][1],  1553 : : setl\_tbl.selector[0][2],  1554 : : setl\_tbl.selector[1][2],  1555 : : setl\_tbl.selector[2][2],  1556 : : setl\_tbl.selector[3][2],  1557 : : setl\_tbl.selector[4][2],  1558 : : setl\_tbl.selector[0][3],  1559 : : setl\_tbl.selector[1][3],  1560 : : setl\_tbl.selector[2][3],  1561 : : setl\_tbl.selector[3][3],  1562 : : setl\_tbl.selector[4][3],  1563 : : setl\_tbl.selector[0][4],  1564 : : setl\_tbl.selector[1][4],  1565 : : setl\_tbl.selector[2][4],  1566 : : setl\_tbl.selector[3][4],  1567 : : setl\_tbl.selector[4][4]);  1568 : :   1569 : : ///Dump TD\_SETL  1570 : :   1571 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1572 : : "setl\_tbl.td.setl\_td\_first",  1573 : : setl\_tbl.td.setl\_td\_first);  1574 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1575 : : "setl\_tbl.td.setl\_td\_gain",  1576 : : setl\_tbl.td.setl\_td\_gain);  1577 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1578 : : "setl\_tbl.td.setl\_td\_offset",  1579 : : setl\_tbl.td.setl\_td\_offset);  1580 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1581 : : "setl\_tbl.td.setl\_td\_max",  1582 : : setl\_tbl.td.setl\_td\_max);  1583 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1584 : : "setl\_tbl.td.setl\_td\_min",  1585 : : setl\_tbl.td.setl\_td\_min);  1586 : :   1587 : :   1588 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1589 : : "setl\_tbl.td.setl\_td\_table.mesh\_size\_x",  1590 : : setl\_tbl.td.setl\_td\_table.mesh\_size\_x);  1591 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1592 : : "setl\_tbl.td.setl\_td\_table.mesh\_size\_y",  1593 : : setl\_tbl.td.setl\_td\_table.mesh\_size\_y);  1594 : :   1595 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1596 : : "setl\_tbl.td.setl\_td\_table.mesh\_x",  1597 : : setl\_tbl.td.setl\_td\_table.mesh\_x);  1598 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1599 : : "setl\_tbl.td.setl\_td\_table.mesh\_y",  1600 : : setl\_tbl.td.setl\_td\_table.mesh\_y);  1601 : :   1602 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1603 : : "setl\_tbl.td.setl\_td\_table.pos\_x",  1604 : : setl\_tbl.td.setl\_td\_table.pos\_x);  1605 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1606 : : "setl\_tbl.td.setl\_td\_table.pos\_y",  1607 : : setl\_tbl.td.setl\_td\_table.pos\_y);  1608 : :   1609 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1610 : : "setl\_tbl.td.setl\_td\_table.data\_byte",  1611 : : setl\_tbl.td.setl\_td\_table.data\_byte);  1612 : : ///End Dumping TD\_SETL  1613 : :   1614 : : ///Dump SIZER\_SETL  1615 : :   1616 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1617 : : "setl\_tbl.sizer.setl\_sizer\_first",  1618 : : setl\_tbl.sizer.setl\_sizer\_first);  1619 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1620 : : "setl\_tbl.sizer.setl\_sizer\_gain",  1621 : : setl\_tbl.sizer.setl\_sizer\_gain);  1622 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1623 : : "setl\_tbl.sizer.setl\_sizer\_offset",  1624 : : setl\_tbl.sizer.setl\_sizer\_offset);  1625 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1626 : : "setl\_tbl.sizer.setl\_sizer\_max",  1627 : : setl\_tbl.sizer.setl\_sizer\_max);  1628 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1629 : : "setl\_tbl.sizer.setl\_sizer\_min",  1630 : : setl\_tbl.sizer.setl\_sizer\_min);  1631 : :   1632 : :   1633 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1634 : : "setl\_tbl.sizer.setl\_sizer\_table.mesh\_size\_x",  1635 : : setl\_tbl.sizer.setl\_sizer\_table.mesh\_size\_x);  1636 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1637 : : "setl\_tbl.sizer.setl\_sizer\_table.mesh\_size\_y",  1638 : : setl\_tbl.sizer.setl\_sizer\_table.mesh\_size\_y);  1639 : :   1640 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1641 : : "setl\_tbl.sizer.setl\_sizer\_table.mesh\_x",  1642 : : setl\_tbl.sizer.setl\_sizer\_table.mesh\_x);  1643 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1644 : : "setl\_tbl.sizer.setl\_sizer\_table.mesh\_y",  1645 : : setl\_tbl.sizer.setl\_sizer\_table.mesh\_y);  1646 : :   1647 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1648 : : "setl\_tbl.sizer.setl\_sizer\_table.pos\_x",  1649 : : setl\_tbl.sizer.setl\_sizer\_table.pos\_x);  1650 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_FLOAT\_PARAMS,  1651 : : "setl\_tbl.sizer.setl\_sizer\_table.pos\_y",  1652 : : setl\_tbl.sizer.setl\_sizer\_table.pos\_y);  1653 : :   1654 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_UINT32\_PARAMS,  1655 : : "setl\_tbl.sizer.setl\_sizer\_table.data\_byte",  1656 : : setl\_tbl.sizer.setl\_sizer\_table.data\_byte);  1657 : : ///End Dumping SIZER\_SETL  1658 : :   1659 : : ///End Dumping SETL TABLE:  1660 : :   1661 : 1063 : Logger::instance()->log(LogMessageTable::eCONF\_DUMP,  1662 : : "\*\*\*\*\*\*\*\*\*\*End Dumping SECOND PARAMS\*\*\*\*\*\*\*\*\*\*\*");  1663 : : ////////////////////////END DUMPING COLUMN PARAMS//////////////////////  1664 : :   1665 : :   1666 : : }  1667 : :   1668 : : ConversionEngine::ConversionEngine()  1669 : 1095 : :cSlidingWindowThreadID( 1024 ),  1670 : : cOutputGenThreadID( 1023 ),  1671 : : mParams( ConverterParams::instance() ),  1672 : : mFixedModeSFTable( 0 ),  1673 : : mWindow(),  1674 : : mFixedModeCellProcs(),  1675 : : mOutputGen(),  1676 : : mThreadManager(),  1677 : : mFixedModeDistributors(),  1678 : : mFixedModeTFDistributor(),  1679 : : mShotOffset(),  1680 : : mBlkUnit(),  1681 : : mShotTimeStep(),  1682 : 1095 : mShotTimeThr() {  1683 : : }  1684 : :   1685 : 2190 : ConversionEngine::~ConversionEngine() {  1686 [ + + ][ # # ]: 1095 : delete mOutputGen;  1687 : : }  1688 : :   1689 : 1215 : } //namespace nft |

|  |
| --- |
| Generated by: [LCOV version 1.9](http://ltp.sourceforge.net/coverage/lcov.php) |