| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [Output/src](http://docs.google.com/index.html) - SChipCnfCreator.cpp (source / [functions](http://docs.google.com/SChipCnfCreator.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 40 | 48 | 83.3 % | | Date: | 2012-03-23 |  | Functions: | 6 | 9 | 66.7 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 1 | 6 | 16.7 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include <stdio.h>  2 : : #include "ErrorHandler.h"  3 : : #include "Logger.h"  4 : : #include "SChipCnfCreator.h"  5 : : #include "UnitConversion.h"  6 : : #include "RetryFunc.h"  7 : :   8 : : namespace nft {  9 : :   10 : 1001 : SChipCnfCreator::SChipCnfCreator() {  11 : : }  12 : :   13 : :   14 : 1001 : SChipCnfCreator::~SChipCnfCreator() {  15 : : }  16 : :   17 : :   18 : 0 : SChipCnfCreator::SChipCnfCreator(const SChipCnfCreator & source) {  19 : : }  20 : :   21 : : SChipCnfCreator &  22 : 0 : SChipCnfCreator::operator=(const SChipCnfCreator & source) {  23 : :   24 : 0 : return \*this;  25 : : }  26 : :   27 : : void  28 : : SChipCnfCreator::dump(const DPMANAGER\_MESSAGE\_CONVERTER \*pStartMsg,   29 : : const sgs\_column\_t \*secParam,  30 : 1001 : const std::string & inFileName) throw(DPL::Error \*) {  31 : :   32 : 1001 : FILE \*fp = NULL;  33 : 1001 : try  34 : : {  35 : 1001 : fp = nftFopenWithRetry(inFileName.c\_str(), "w");  36 : : }  37 [ # # ]: 0 : catch(DPL::Error \*pChild)   38 : : {  39 : 0 : fp = NULL;  40 : : }  41 : : //#ifdef ERROR\_SEED //<<Test\_Id:SchipCnfCreator\_fp\_Set\_To\_NULL>>  42 : : // fp = NULL;  43 : : //#endif  44 [ - + ]: 1001 : if(NULL == fp)  45 : : {  46 : 0 : Logger::instance()->log(LogMessageTable::eDATA\_READER\_FILE\_CREATE\_FAIL,  47 : : inFileName.c\_str());  48 : :   49 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  50 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  51 : : ,ErrorHandler::eErrFileOpen  52 : : ,pErr->getErrorMsg((ErrorHandler::eErrFileOpen)  53 : : ,inFileName.c\_str()));  54 : : }  55 : :   56 : 1001 : fprintf(fp," Format.Version : \"SDF-12i R0.81\"");  57 : : //fprintf(fp,"\n Vchip.Name : %s",);  58 : 1001 : fprintf(fp,"\n Drawing.Mode: %u", pStartMsg->usProcessingMode);  59 : 1001 : fprintf(fp,"\n Address.Unit : %u", pStartMsg->uiAU);  60 : 1001 : fprintf(fp,"\n Mask.Center.x : %lu", secParam->mask.mask\_center\_x);  61 : 1001 : fprintf(fp,"\n Mask.Center.y : %lu", secParam->mask.mask\_center\_y);  62 : 1001 : fprintf(fp,"\n Vchip.Origin.x : %lld", pStartMsg->llVchipOriginX );  63 : 1001 : fprintf(fp,"\n Vchip.Origin.y : %lld", pStartMsg->llVchipOriginY );  64 : 1001 : fprintf(fp,"\n Vchip.Size.x: %llu", pStartMsg->ullVchipSizeX );  65 : 1001 : fprintf(fp,"\n Vchip.Size.y: %llu", pStartMsg->ullVchipSizeY );  66 : 1001 : fprintf(fp,"\n Stripe.Height : %llu", pStartMsg->ullDpbHeight);  67 : 1001 : fprintf(fp,"\n SF.Size.x : %d", pStartMsg->iSFSizeX);  68 : 1001 : fprintf(fp,"\n SF.Size.y : %d", pStartMsg->iSFSizeY);  69 : 1001 : fprintf(fp,"\n TF.Size.x : %d", pStartMsg->iTFSizeX);  70 : 1001 : fprintf(fp,"\n TF.Size.y : %d", pStartMsg->iTFSizeY);  71 : 1001 : fprintf(fp,"\n Vchip.Stripe.Layers : %u", pStartMsg->uiSTLCountInColumn);  72 : 1001 : fprintf(fp,"\n Vchip.SF.Layers : %u", pStartMsg->uiSFLCountInColumn);  73 : 1001 : fprintf(fp,"\n Laser.Wavelength : %lf", secParam->unit.laser\_wavelength);  74 : 1001 : fprintf(fp,"\n Laser.Count : %u", secParam->unit.laser\_count);  75 : 1001 : fprintf(fp,"\n Swing.Back.1.x : %d", pStartMsg->iSwingBackOffsetX1);  76 : 1001 : fprintf(fp,"\n Swing.Back.1.y : %d", pStartMsg->iSwingBackOffsetY1);  77 : 1001 : fprintf(fp,"\n Swing.Back.2.x : %d", pStartMsg->iSwingBackOffsetX2);  78 : 1001 : fprintf(fp,"\n Swing.Back.2.y : %d", pStartMsg->iSwingBackOffsetY2);  79 : 1001 : fprintf(fp,"\n Swing.Back.3.x : %d", pStartMsg->iSwingBackOffsetX3);  80 : 1001 : fprintf(fp,"\n Swing.Back.3.y : %d", pStartMsg->iSwingBackOffsetY3);  81 : 1001 : fprintf(fp,"\n Swing.Back.4.x : %d", pStartMsg->iSwingBackOffsetX4);  82 : 1001 : fprintf(fp,"\n Swing.Back.4.y : %d", pStartMsg->iSwingBackOffsetY4);  83 : 1001 : fprintf(fp,"\n Swing.Back.5.x : %d", pStartMsg->iSwingBackOffsetX5);  84 : 1001 : fprintf(fp,"\n Swing.Back.5.y : %d", pStartMsg->iSwingBackOffsetY5);  85 : 1001 : fprintf(fp,"\n Total.Shot.Cycle.Rate : %lf",  86 : : pStartMsg->dShotCycleTimeScaleFactor);  87 : 1001 : fprintf(fp,"\n GMC.Max.Allowable.Value : %u\n",  88 : : secParam->gmc.max\_allowable\_value);  89 : :   90 [ # # ]: 1001 : nftFcloseWithRetry(fp);  91 : : }  92 : :   93 : 1215 : }//namespace nft |

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