

FSP TCL Reference

Product Version 23.1
September 2023

© 2023 Cadence Design Systems, Inc.
Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Open SystemC, Open SystemC Initiative, OSCI, SystemC, and SystemC Initiative are trademarks or registered trademarks of Open SystemC Initiative, Inc. in the United States and other countries and are used with permission.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information. Cadence is committed to using respectful language in our code and communications. We are also active in the removal and replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

1	19
FSP TCL Commands	19
AddCustomMenuCommand	29
AddDecap	30
AddDecapOrCAD	31
AddInstanceCustomAttribute	32
AddInstancePinCustomAttribute	32
AddNet	33
AddNetCustomAttribute	34
AddPart	34
AddPartModel	35
AddPartOrCAD	36
AddPowerRegulator	37
AddRulesFilePath	38
AddSearchNReplaceNetNamePattern	39
AddSecondary	39
AddTermination	40
AllocatePairPinsTogether	40
ArchiveProject	41
AutoAddPowerRegulators	41
AutoMapPowerRegulators	42
AutoNetGroupDesignGroupWise	43
AutoNetGroupDUTBasedOnConnectedDevices	44
AutoNetGroupGroupWise	44
AutoNetGroupInterface	45
AutoNetGroupSwappablePins	45
AutoSplitSymbol	46
AutoUpdateAllDeviceFPGAPortNames	47
AutoUpdateDeviceFPGAPortNames	47
BoolTest	48
ChangeGlobalOrCADLibraryPath	48
ChangeNetName	49
ChangeProtocolOrder	50

ChangeRules	50
ChangeSchematicsSymbolFileReference	51
CheckDesignConsistency	51
ClearAllMessageWindows	52
ClearNets	52
CloseProject	53
ConvertLRFTToRealPart	53
ConvertLRFTToRealPartOrCAD	54
ConvertVIToRealInterface	55
ConvertVIToRealInterfaceOrCAD	56
CreateCommonGroup	57
CreateNewDEHDLProject	57
CreateNewNetGroup	58
CreateNewOrCADProject	59
CreateNewProject	59
CreatePartFromCSV	60
CreateProtocolFromConstraintsPinoutfile	61
CreateProtocolFromCSV	61
CreateProtocolFromExistingProtocol	62
CreateProtocolFromLibraryModel	63
CreateSystemAceChain	64
CreateTargetDeviceSet	64
CreateVIFromCSV	65
CreateVIFromExistingVI	66
CreateVIFromLibraryModel	66
CreateVIFromStringList	67
CreateVirtualInterfaceFromConstraintsPinoutfile	67
CreateVirtualInterfaceFromXDCfile	68
CreateWideBus	69
DeferSpecialPurposePinsUsage	69
DefineJTAGChain	70
DefinePROMChain	71
DeleteAllDesignNetGroups	71
DeleteAllInstanceCustomAttributes	72
DeleteAllInstancePinCustomAttributes	72
DeleteAllNetCustomAttributes	73
DeleteInstance	73

DeleteInstanceCustomAttribute	74
DeleteInstanceNets	75
DeleteInstancePinCustomAttribute	75
DeleteInstanceSelectedPinNets	76
DeleteJTAGChain	76
DeleteNetCustomAttribute	77
DeleteNetGroups	77
DeleteNets	78
DeletePowerRegulator	79
DeletePowerRegulators	80
DeletePROMChain	81
DeleteTermination	81
DifferSpecialPurposePinsUsage	82
DoubleListTest	82
DoubleTest	83
env	84
ExportConstraints	84
ExportCSVFromConnectorMapping	85
ExportCSVfromDesignConnectivity	85
ExportCSVfromDesignExplorer	86
ExportCSVFromFPGAPort	87
ExportCSVFromInstancePart	88
ExportCSVFromPart	88
ExportCSVFromProtocol	89
ExportCSVFromVI	90
ExportCSVSchematicSymbolEditor	90
ExportDecaps	91
ExportDesignBlockSymbolCSV	91
ExportDesignForPinSwap	92
ExportDeviceConstraints	92
ExportPDF	93
ExportPinAssignemntsForConnector	94
ExportPinAssignmentsForConnector	94
ExportPlacementData	95
ExportProtocolDefinition	95
ExportSpreadSheetFiles	96
ExportVirtualInterfaceDefinition	96

FlipInstance	97
GenerateAllegroSymbol	98
GenerateASADesign	98
GenerateConstraintFiles	99
GenerateDEHDL Schematics	99
GenerateDesignBlockSymbol	100
GenerateLayoutData	101
GenerateNCReport	102
GenerateNCReportStatus	102
GenerateNetgroupRatbundle	103
GenerateNetList	103
GenerateOrCAD Schematics	104
GenerateOrCADSymbol	105
GeneratePinNumberMappingFile	105
GeneratePlanAheadScripts	106
GenerateVerilogBrdDescFile	106
GetAllContiguousSignal	107
GetAllDoNotConnect	108
GetAllegroCPMFileName	108
GetAllLockedNetNames	109
GetAllProcessOptionNames	109
GetAllTargetDevice	110
GetAllUseBanks	110
GetApplyNetNameTemplate	111
GetAvailableIOPinCount	112
GetAvailableIOPinNumberList	112
GetAvailableLicenses	113
GetAvailableNetGroupNames	113
GetAvaliableLicenseType	114
GetBankAvailableIOPinCount	114
GetBankAvailableIOPinNumberList	115
GetBankIOPinCount	116
GetBankIOPinNumberList	116
GetBankName	117
GetBanksNameList	118
GetBankVCCOlevel	118
GetBankVREFlevel	119

GetBoardDimensionUnits	119
GetBoardHeight	120
GetBoardWidth	120
GetCaptureINIFilePath	121
GetClockBuffer	121
GetConnectedInstancesList	122
GetConnectedNetGroupNames	122
GetConnectedPinCount	123
GetContiguousSignal	123
GetCurrentLicenseType	124
GetDeepNWideGroupInstanceList	125
GetDeepNWideGroups	125
GetDehdlFPGAHierBlockLibrayName	126
GetDesignDecapLibrarySymbolNameList	126
GetDesignTopBlockLibAndName	127
GetDeviceFamilyName	127
GetDeviceInstanceList	128
GetDiffPairPin	128
GetDoNotConnect	129
GetDontUseBanks	129
GetDontUseBanksForProtocol	130
GetDraDirectoriesPaths	131
GetDraPath	132
GetEnvVariable	132
GetEnvVariables	133
GetFESymbolMapping	133
GetFPGAHierBlockLibAndName	134
GetFPGAPinMappingDirectoriesPaths	134
GetFPGAPinMappingPath	135
GetFSPBlockLibAndName	136
GetFSPPartName	136
GetGenerateSymbolLibraryName	137
GetGroupNameList	137
GetGroupOrBankPinNameList	138
GetGroupOrBankPinNumberList	138
GetInstanceCompletePartName	139
GetInstanceCustomAttributeValue	139

GetInstanceDRAbsoluteFilePath	140
GetInstanceFootprint	141
GetInstanceHeight	141
GetInstanceMappingAbsoluteFilePath	142
GetInstanceNameList	143
GetInstanceNamesOfJTAGChain	143
GetInstancePartName	144
GetInstancePinCustomAttributeValue	144
GetInstanceRotation	145
GetInstanceRulesAbsoluteFilePath	145
GetInstanceSide	146
GetInstanceWidth	147
GetInstanceXCoordinate	147
GetInstanceYCoordinate	148
GetInstanceZOrderValue	149
GetInterfaceInstanceList	149
GetIOPinCount	150
GetIOPinNumberList	150
GetIsolateStatus	151
GetJedecType	151
GetJTAGChains	152
GetMaximumNetGroupSize	152
GetMaxOutputsPerBank	153
GetMGTPairPin	153
GetNetGroupSize	154
GetNetName	154
GetNetNamesOfNetGroup	155
GetNetType	156
GetNoOfConnectedNetGroups	156
GetNoOfNetGroups	157
GetNotConnectedPinCount	157
GetOutputDirPath	158
GetPartCustomAttributeValue	158
GetPartDimensionUnit	159
GetPartHeight	159
GetPartPinCustomAttribValue	160
GetPartPinXCoordinate	160

GetPartPinYCoordinate	161
GetPartWidth	162
GetPartXOffset	162
GetPartYOffset	163
GetPCBOutlineHeight	163
GetPCBOutlineSettings	164
GetPCBOutlineWidth	165
GetPinName	165
GetPinNameList	166
GetPinNumber	167
GetPinNumberList	167
GetPinNumbersOfNetGroup	168
GetPinUseType	168
GetPinXCoordinate	169
GetPinYCoordinate	170
GetPowerPinsList	170
GetPowerRegulator	171
GetPowerRegulatorName	172
GetPowerRegulators	173
GetPowerRegulatorVoltage	173
GetPreservePins	174
GetProjectFilePath	174
GetProjectName	175
GetProtocolNames	176
GetRegulatorName	176
GetReleasePath	177
GetResolvedRulesFilePaths	178
GetResources	178
GetRuleFilePath	179
GetRulesFilePaths	180
GetRulesWorkingDir	180
GetSchematicBoardFileName	181
GetSchematicBoardFilePath	181
GetSchematicsEnvironment	182
GetSchematicsSymbolFileReference	182
GetSearchNReplaceNetNamePattern	183
GetSnapShotTimeInterval	183

GetSupportedFamilyNames	184
GetSwappablePins	184
GetSymbolLibraryName	185
GetSymbolPartName	186
GetTargetDevice	186
GetTargetFPGAFamilies	187
GetTargetInstanceListForDevice	188
GetTerminationNames	188
GetTerminationSymbol	189
GetUseBanks	190
GetUseBanksForProtocol	191
GetVectorCloseBrace	191
GetVectorOpenBrace	192
GetViiInstanceList	192
GetVirtualInterfaceNames	193
GetVoltageLevel	193
GetWorkingDir	194
HideAllNets	195
HideInstanceNets	195
ImportConstraints	196
ImportCSVInDesignConnectivity	196
ImportCSVInDesignExplorer	197
ImportDecaps	198
ImportFromAllegroBoard	199
ImportInstanceConstraints	199
ImportPDC	200
ImportPinAssignmentsForConnector	201
ImportPlacementXMLFile	201
ImportPowerMappingData	202
InitDesignNetNameDatabase	202
InitInstancePinLocation	203
IntListTest	203
IntTest	204
IsCheckSymbolLargerThanPageBorder	205
IsDraExist	205
isECOMode	206
IsFPGAPinMappingFileExist	206

IsGenerateSFReport	207
IsNetGroupConnected	208
IsOptimizeTDConnectorUtilization	208
IsPerformSecondPassOptimization	209
IsPinNameASNetName	209
IsSkipConnectedPinsInContiguousConnections	210
IsSkipConnectedPinsInContiguousConnectionsForProtocol	210
IsUsePart	211
LinkToFESymbol	211
LinkToFESymbolOrCAD	212
LoadProcessOptions	213
LoadWorkFlow	213
LockInstanceNets	214
LockNets	214
MapConnectorPinAssignment	215
MapPortNamestoPinNames	216
MapPortNameToNetName	217
MapPowerFilterToInstancePin	217
MapResources	218
MapTermination	219
MapTerminationOrCAD	219
MapTerminationToInstancePin	220
MapTerminationToInstancePinOtherEnd	221
MergeAllSymbolSplits	222
MoveDeviceNet	222
MoveInstance	223
MovePinNet	223
NewProject	224
OpenDEHDLProject	225
OpenProject	225
PlaceInstance	226
PreservePairPins	226
PreserveTrueDifferentialPins	227
Quit	227
RemoveAllProcessOptions	228
RemoveAllProtocols	229
RemoveDeepNWideGroup	229

RemoveProcessOption	230
RemoveProtocol	230
RemoveRulesFilePath	231
RemoveSearchNReplaceNetNamePatterns	232
RenameInstance	232
RenameNetGroup	233
RenamePowerRegulator	233
RenameProcessOption	234
ReOptimizeProtocol	235
ReplaceFPGA	236
ReportAllDRAFiles	236
ReportAllFPGAPinMappingFiles	237
ReportAllMappingFiles	238
ReportAllRulesFiles	238
ReportDesign	239
ReportDesignFileReferences	239
ReportDesignTermination	240
ReportDesignToFile	240
ReportInstance	241
ReportInstanceToFile	241
ResetAllDoNotConnect	242
ResetAllTerminationsRefDes	243
ResetAssignedToPins	243
ResetCachedNetNames	244
ResetDesignPowerMapping	245
ResetDoNotConnectPin	245
ResetExternPin	246
ResetFSPRegistry	246
ResetInstancePowerMapping	247
ResetMappedPorts	247
ResetPDCPreservedVREFs	248
ResetPowerMapping	249
ResetPreserveUnusedPinsInBank	249
ResetResources	250
ResetSpecifyNetNames	250
ResetUsePins	251
RotateInstance	251

RunDesign	252
RunDesignWithRunSet	253
RunInstance	253
SaveProcessOptions	254
SaveProject	255
SaveProjectAs	255
SaveWorkFlowAs	256
SetAllegroCPMFileName	256
SetAutoNetGroupInterface	257
SetAutoNetGroupProtocol	258
SetBoardDimensionUnits	258
SetBoardHeight	259
SetBoardWidth	259
SetCachedNetNames	260
SetCaptureINIFilePath	261
SetClockBuffer	261
SetContiguousSignal	262
SetContiguousSignalForProtocol	263
SetDehdlFPGAHierBlockLibrayName	263
SetDesignConnectivityView	264
SetDesignExplorerView	264
SetDesignNetNameTemplate	265
SetDesignProtocolNetNameTemplate	266
SetDesignTopBlockLibAndName	267
SetDevicePreservePins	267
SetDoNotCombineDifferentVoltageInputsintoSameBank	268
SetDoNotConnectPin	268
SetDontUseBanks	269
SetDontUseBanksForProtocol	270
SetEnvVariable	270
SetExternPin	271
SetFPGAHierBlockLibAndName	272
SetFSPBlockLibAndName	272
SetGenerateSymbolLibraryName	273
SetIsCheckSymbolLargerThanPageBorder	273
SetIsFilterLogMessages	274
SetIsGenerateSFReport	274

SetIsolateStatus	275
SetIsPerformSecondPassOptimization	275
SetLicenseType	276
SetMaximumNetGroupSize	277
SetMaxOutputsPerBank	277
SetOutputDirPath	278
SetPCBOutlineHeight	278
SetPCBOutlineSettings	279
SetPCBOutlineWidth	279
SetPinNameASNetName	280
SetPowerRegulator	281
SetPowerRegulatorVoltage	281
SetPreservePins	282
SetPreserveUnusedPinsInBank	283
SetRulesWorkingDir	283
SetSchematicBoardFileName	284
SetSchematicBoardFilePath	284
SetSearchNReplaceNetNamePattern	285
SetSkipConnectedPins	285
SetSnapShotTimeInterval	286
SetSymbolPinDirection	287
SetUseBanks	287
SetUseBanksForMultiDevices	288
SetUseBanksForProtocol	289
SetVoltageForMultiVoltagePins	290
SpecifyDiffPinTermination	290
SpecifyNetNames	291
SpecifyOtherEndTermination	292
SplitSymbolBankWise	292
SplitSymbolConnectionWise	293
StringListTest	293
StringStringListTest	294
StringStringMapTest	295
StringTest	295
SwapGroups	296
TargetDevice	297
TargetToMultipleDevices	298

ToggleOptimizeTDConnectorUtilization	298
ToggleSecondPassOptimization	299
UnLockInstanceNets	299
UnlockNets	300
UnPreservePairPins	301
UnPreserveTrueDifferentialPins	302
UpdateAllegroSchematics	302
UpdateAssignedToPinsWithConnectedPins	303
UpdateDeepNWideGroupName	303
UpdateDesignFromAllegro	304
UpdateDeviceDataBase	305
UpdateFPGAPortsFromCSV	305
UpdateInstanceFootprint	306
UpdateInstanceLocation	307
UpdateInstancePartFromCSV	307
UpdateInstanceSymbolFromCSV	308
UpdateLayoutData	309
UpdateNetGroup	310
UpdateOrCADSchematics	310
UpdatePartDescription	311
UpdatePartFromCSV	311
UpdateProtocolFromCSV	312
UpdateVIFromCSV	313
UpRevDesignDatabase	314
UpRevLibraryPartDatabase	315
Version	315
ZoomFitAll	316
ZoomFitInstance	316
2	317
FlipAllDecapPortMapping	317
Return	317
Syntax	317
Parameters	317
Examples	317
Related Commands	317
3	318

GenerateVirtualComponentPinMappingFile	318
Return	318
Syntax	318
Parameters	318
Examples	318
Related Commands	318
4	319
GetBusNamesOfGroup	319
Return	319
Syntax	319
Parameters	319
Examples	319
Related Commands	319
5	320
GetMultiTargetDevices	320
Return	320
Syntax	320
Parameters	320
Examples	320
Related Commands	320
6	321
GetMultiTargettedGroupUseBanks	321
Return	321
Syntax	321
Parameters	321
Examples	321
Related Commands	322
7	323
GetPinNamesOfBus	323
Return	323
Syntax	323
Parameters	323
Examples	323
Related Commands	323

8	324
GetTargetPinFunction	324
Return	324
Syntax	324
Parameters	324
Examples	324
Related Commands	324
9	325
IsGroupTargettedToDevice	325
Return	325
Syntax	325
Parameters	325
Examples	325
Related Commands	325
10	326
IsGroupTargettedToMultipleDevices	326
Return	326
Syntax	326
Parameters	326
Examples	326
Related Commands	326
11	327
IsInterfacePrtocolGroup	327
Return	327
Syntax	327
Parameters	327
Examples	327
Related Commands	327
12	328
RenameProtocol	328
Return	328
Syntax	328
Parameters	328
Examples	328

Related Commands	329
13	330
ReportCdsLibLibraryPaths	330
Return	330
Syntax	330
Parameters	330
Examples	330
Related Commands	330
14	331
ResetTargetPinFunction	331
Return	331
Syntax	331
Parameters	331
Examples	331
Related Commands	331
15	332
SetTargetPinFunction	332
Return	332
Syntax	332
Parameters	332
Examples	332
Related Commands	333

FSP TCL Commands

- [AddCustomMenuCommand](#)
- [AddDecap](#)
- [AddDecapOrCAD](#)
- [AddInstanceCustomAttribute](#)
- [AddInstancePinCustomAttribute](#)
- [AddNet](#)
- [AddNetCustomAttribute](#)
- [AddPart](#)
- [AddPartModel](#)
- [AddPartOrCAD](#)
- [AddPowerRegulator](#)
- [AddRulesFilePath](#)
- [AddSearchNReplaceNetNamePattern](#)
- [AddSecondary](#)
- [AddTermination](#)
- [AllocatePairPinsTogether](#)
- [ArchiveProject](#)
- [AutoAddPowerRegulators](#)
- [AutoMapPowerRegulators](#)
- [AutoNetGroupDesignGroupWise](#)
- [AutoNetGroupDUTBasedOnConnectedDevices](#)
- [AutoNetGroupGroupWise](#)
- [AutoNetGroupInterface](#)
- [AutoNetGroupSwappablePins](#)
- [AutoSplitSymbol](#)
- [AutoUpdateAllDeviceFPGAPortNames](#)
- [AutoUpdateDeviceFPGAPortNames](#)
- [BoolTest](#)
- [ChangeGlobalOrCADLibraryPath](#)
- [ChangeNetName](#)
- [ChangeProtocolOrder](#)
- [ChangeRules](#)
- [ChangeSchematicsSymbolFileReference](#)
- [CheckDesignConsistency](#)
- [ClearAllMessageWindows](#)
- [ClearNets](#)
- [CloseProject](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)

- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)
- [CreateCommonGroup](#)
- [CreateNewDEHDLProject](#)
- [CreateNewNetGroup](#)
- [CreateNewOrCADProject](#)
- [CreateNewProject](#)
- [CreatePartFromCSV](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [CreateProtocolFromCSV](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromLibraryModel](#)
- [CreateSystemAceChain](#)
- [CreateTargetDeviceSet](#)
- [CreateVIFromCSV](#)
- [CreateVIFromExistingVI](#)
- [CreateVIFromLibraryModel](#)
- [CreateVIFromStringList](#)
- [CreateVirtualInterfaceFromConstraintsPinoutfile](#)
- [CreateVirtualInterfaceFromXDCfile](#)
- [CreateWideBus](#)
- [DeferSpecialPurposePinsUsage](#)
- [DefineJTAGChain](#)
- [DefinePROMChain](#)
- [DeleteAllDesignNetGroups](#)
- [DeleteAllInstanceCustomAttributes](#)
- [DeleteAllInstancePinCustomAttributes](#)
- [DeleteAllNetCustomAttributes](#)
- [DeleteInstance](#)
- [DeleteInstanceCustomAttribute](#)
- [DeleteInstanceNets](#)
- [DeleteInstancePinCustomAttribute](#)
- [DeleteInstanceSelectedPinNets](#)
- [DeleteJTAGChain](#)
- [DeleteNetCustomAttribute](#)
- [DeleteNetGroups](#)
- [DeleteNets](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [DeletePROMChain](#)
- [DeleteTermination](#)
- [DifferSpecialPurposePinsUsage](#)
- [DoubleListTest](#)

- [DoubleTest](#)
- [env](#)
- [ExportConstraints](#)
- [ExportCSVFromConnectorMapping](#)
- [ExportCSVfromDesignConnectivity](#)
- [ExportCSVfromDesignExplorer](#)
- [ExportCSVFromFPGAPort](#)
- [ExportCSVFromInstancePart](#)
- [ExportCSVFromPart](#)
- [ExportCSVFromProtocol](#)
- [ExportCSVFromVI](#)
- [ExportCSVSchematicSymbolEditor](#)
- [ExportDecaps](#)
- [ExportDesignBlockSymbolCSV](#)
- [ExportDesignForPinSwap](#)
- [ExportDeviceConstraints](#)
- [ExportPDF](#)
- [ExportPinAssignemntsForConnector](#)
- [ExportPinAssignmentsForConnector](#)
- [ExportPlacementData](#)
- [ExportProtocolDefinition](#)
- [ExportSpreadSheetFiles](#)
- [ExportVirtualInterfaceDefinition](#)
- [FlipInstance](#)
- [GenerateAllegroSymbol](#)
- [GenerateASADesign](#)
- [GenerateConstraintFiles](#)
- [GenerateDEHDL Schematics](#)
- [GenerateDesignBlockSymbol](#)
- [GenerateLayoutData](#)
- [GenerateNCReport](#)
- [GenerateNCReportStatus](#)
- [GenerateNetgroupRatbundle](#)
- [GenerateNetList](#)
- [GenerateOrCADSchematics](#)
- [GenerateOrCADSymbol](#)
- [GeneratePinNumberMappingFile](#)
- [GeneratePlanAheadScripts](#)
- [GenerateVerilogBrdDescFile](#)
- [GetAllContiguousSignal](#)
- [GetAllDoNotConnect](#)
- [GetAllegroCPMFileName](#)
- [GetAllLockedNetNames](#)

- [GetAllProcessOptionNames](#)
- [GetAllTargetDevice](#)
- [GetAllUseBanks](#)
- [GetApplyNetNameTemplate](#)
- [GetAvailableIOPinCount](#)
- [GetAvailableIOPinNumberList](#)
- [GetAvailableLicenses](#)
- [GetAvailableNetGroupName](#)
- [GetAvailableLicenseType](#)
- [GetBankAvailableIOPinCount](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinCount](#)
- [GetBankIOPinNumberList](#)
- [GetBankName](#)
- [GetBanksNameList](#)
- [GetBankVCCOlevel](#)
- [GetBankVREFlevel](#)
- [GetBoardDimensionUnits](#)
- [GetBoardHeight](#)
- [GetBoardWidth](#)
- [GetCaptureINIFilePath](#)
- [GetClockBuffer](#)
- [GetConnectedInstancesList](#)
- [GetConnectedNetGroupNames](#)
- [GetConnectedPinCount](#)
- [GetContiguousSignal](#)
- [GetCurrentLicenseType](#)
- [GetDeepNWideGroupInstanceList](#)
- [GetDeepNWideGroups](#)
- [GetDehdlFPGAHierBlockLibrayName](#)
- [GetDesignDecapLibrarySymbolNameList](#)
- [GetDesignTopBlockLibAndName](#)
- [GetDeviceFamilyName](#)
- [GetDeviceInstanceList](#)
- [GetDiffPairPin](#)
- [GetDoNotConnect](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [GetDraDirectoriesPaths](#)
- [GetDraPath](#)
- [GetEnvVariable](#)
- [GetEnvVariables](#)
- [GetFESymbolMapping](#)

- [GetFPGAHierBlockLibAndName](#)
- [GetFPGAPinMappingDirectoriesPaths](#)
- [GetFPGAPinMappingPath](#)
- [GetFSPBlockLibAndName](#)
- [GetFSPPartName](#)
- [GetGenerateSymbolLibraryName](#)
- [GetGroupNameList](#)
- [GetGroupOrBankPinNameList](#)
- [GetGroupOrBankPinNumberList](#)
- [GetInstanceCompletePartName](#)
- [GetInstanceCustomAttributeValue](#)
- [GetInstanceDRAbsoluteFilePath](#)
- [GetInstanceFootprint](#)
- [GetInstanceHeight](#)
- [GetInstanceMappingAbsoluteFilePath](#)
- [GetInstanceNameList](#)
- [GetInstanceNamesOfJTAGChain](#)
- [GetInstancePartName](#)
- [GetInstancePinCustomAttributeValue](#)
- [GetInstanceRotation](#)
- [GetInstanceRulesAbsoluteFilePath](#)
- [GetInstanceSide](#)
- [GetInstanceWidth](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetInstanceZOrderValue](#)
- [GetInterfaceInstanceList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)
- [GetIsolateStatus](#)
- [GetJedecType](#)
- [GetJTAGChains](#)
- [GetMaximumNetGroupSize](#)
- [GetMaxOutputsPerBank](#)
- [GetMGTPairPin](#)
- [GetNetGroupSize](#)
- [GetNetName](#)
- [GetNetNamesOfNetGroup](#)
- [GetNetType](#)
- [GetNoOfConnectedNetGroups](#)
- [GetNoOfNetGroups](#)
- [GetNotConnectedPinCount](#)
- [GetOutputDirPath](#)

- [GetPartCustomAttributeValue](#)
- [GetPartDimensionUnit](#)
- [GetPartHeight](#)
- [GetPartPinCustomAttribValue](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetPartWidth](#)
- [GetPartXOffset](#)
- [GetPartYOffset](#)
- [GetPCBOutlineHeight](#)
- [GetPCBOutlineSettings](#)
- [GetPCBOutlineWidth](#)
- [GetPinName](#)
- [GetPinNameList](#)
- [GetPinNumber](#)
- [GetPinNumberList](#)
- [GetPinNumbersOfNetGroup](#)
- [GetPinUseType](#)
- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetPowerPinsList](#)
- [GetPowerRegulator](#)
- [GetPowerRegulatorName](#)
- [GetPowerRegulators](#)
- [GetPowerRegulatorVoltage](#)
- [GetPreservePins](#)
- [GetProjectFilePath](#)
- [GetProjectName](#)
- [GetProtocolNames](#)
- [GetRegulatorName](#)
- [GetReleasePath](#)
- [GetResolvedRulesFilePaths](#)
- [GetResources](#)
- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [GetRulesWorkingDir](#)
- [GetSchematicBoardFileName](#)
- [GetSchematicBoardFilePath](#)
- [GetSchematicsEnvironment](#)
- [GetSchematicsSymbolFileReference](#)
- [GetSearchNReplaceNetNamePattern](#)
- [GetSnapShotTimeInterval](#)
- [GetSupportedFamilyNames](#)

- [GetSwappablePins](#)
- [GetSymbolLibraryName](#)
- [GetSymbolPartName](#)
- [GetTargetDevice](#)
- [GetTargetFPGAFamilies](#)
- [GetTargetInstanceListForDevice](#)
- [GetTerminationNames](#)
- [GetTerminationSymbol](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetVectorCloseBrace](#)
- [GetVectorOpenBrace](#)
- [GetVlInstanceList](#)
- [GetVirtualInterfaceNames](#)
- [GetVoltageLevel](#)
- [GetWorkingDir](#)
- [HideAllNets](#)
- [HideInstanceNets](#)
- [ImportConstraints](#)
- [ImportCSVInDesignConnectivity](#)
- [ImportCSVInDesignExplorer](#)
- [ImportDecaps](#)
- [ImportFromAllegroBoard](#)
- [ImportInstanceConstraints](#)
- [ImportPDC](#)
- [ImportPinAssignmentsForConnector](#)
- [ImportPlacementXMLFile](#)
- [ImportPowerMappingData](#)
- [InitDesignNetNameDatabase](#)
- [InitInstancePinLocation](#)
- [IntListTest](#)
- [IntTest](#)
- [IsCheckSymbolLargerThanPageBorder](#)
- [IsDraExist](#)
- [isECOMode](#)
- [IsFPGAPinMappingFileExist](#)
- [IsGenerateSFReport](#)
- [IsNetGroupConnected](#)
- [IsOptimizeTDConnectorUtilization](#)
- [IsPerformSecondPassOptimization](#)
- [IsPinNameASNetName](#)
- [IsSkipConnectedPinsInContiguousConnections](#)
- [IsSkipConnectedPinsInContiguousConnectionsForProtocol](#)

- [IsUsePart](#)
- [LinkToFESymbol](#)
- [LinkToFESymbolOrCAD](#)
- [LoadProcessOptions](#)
- [LoadWorkFlow](#)
- [LockInstanceNets](#)
- [LockNets](#)
- [MapConnectorPinAssignment](#)
- [MapPortNamestoPinNames](#)
- [MapPortNameToNetName](#)
- [MapPowerFilterToInstancePin](#)
- [MapResources](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [MapTerminationToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MergeAllSymbolSplits](#)
- [MoveDeviceNet](#)
- [MoveInstance](#)
- [MovePinNet](#)
- [NewProject](#)
- [OpenDEHDLProject](#)
- [OpenProject](#)
- [PlaceInstance](#)
- [PreservePairPins](#)
- [PreserveTrueDifferentialPins](#)
- [Quit](#)
- [RemoveAllProcessOptions](#)
- [RemoveAllProtocols](#)
- [RemoveDeepNWideGroup](#)
- [RemoveProcessOption](#)
- [RemoveProtocol](#)
- [RemoveRulesFilePath](#)
- [RemoveSearchNReplaceNetNamePatterns](#)
- [RenameInstance](#)
- [RenameNetGroup](#)
- [RenamePowerRegulator](#)
- [RenameProcessOption](#)
- [ReOptimizeProtocol](#)
- [ReplaceFPGA](#)
- [ReportAllDRAFiles](#)
- [ReportAllFPGAPinMappingFiles](#)
- [ReportAllMappingFiles](#)

- [ReportAllRulesFiles](#)
- [ReportDesign](#)
- [ReportDesignFileReferences](#)
- [ReportDesignTermination](#)
- [ReportDesignToFile](#)
- [ReportInstance](#)
- [ReportInstanceToFile](#)
- [ResetAllDoNotConnect](#)
- [ResetAllTerminationsRefDes](#)
- [ResetAssignedToPins](#)
- [ResetCachedNetNames](#)
- [ResetDesignPowerMapping](#)
- [ResetDoNotConnectPin](#)
- [ResetExternPin](#)
- [ResetFSPRegistry](#)
- [ResetInstancePowerMapping](#)
- [ResetMappedPorts](#)
- [ResetPDCPreservedVREFs](#)
- [ResetPowerMapping](#)
- [ResetPreserveUnusedPinsInBank](#)
- [ResetResources](#)
- [ResetSpecifyNetNames](#)
- [ResetUsePins](#)
- [RotateInstance](#)
- [RunDesign](#)
- [RunDesignWithRunSet](#)
- [RunInstance](#)
- [SaveProcessOptions](#)
- [SaveProject](#)
- [SaveProjectAs](#)
- [SaveWorkFlowAs](#)
- [SetAllegroCPMFileName](#)
- [SetAutoNetGroupInterface](#)
- [SetAutoNetGroupProtocol](#)
- [SetBoardDimensionUnits](#)
- [SetBoardHeight](#)
- [SetBoardWidth](#)
- [SetCachedNetNames](#)
- [SetCaptureNIFilePath](#)
- [SetClockBuffer](#)
- [SetContiguousSignal](#)
- [SetContiguousSignalForProtocol](#)
- [SetDehdlFPGAHierBlockLibraryName](#)

- [SetDesignConnectivityView](#)
- [SetDesignExplorerView](#)
- [SetDesignNetNameTemplate](#)
- [SetDesignProtocolNetNameTemplate](#)
- [SetDesignTopBlockLibAndName](#)
- [SetDevicePreservePins](#)
- [SetDoNotCombineDifferentVoltageInputsintoSameBank](#)
- [SetDoNotConnectPin](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)
- [SetEnvVariable](#)
- [SetExternPin](#)
- [SetFPGAHierBlockLibAndName](#)
- [SetFSPBlockLibAndName](#)
- [SetGenerateSymbolLibraryName](#)
- [SetIsCheckSymbolLargerThanPageBorder](#)
- [SetIsFilterLogMessages](#)
- [SetIsGenerateSFReport](#)
- [SetIsolateStatus](#)
- [SetIsPerformSecondPassOptimization](#)
- [SetLicenseType](#)
- [SetMaximumNetGroupSize](#)
- [SetMaxOutputsPerBank](#)
- [SetOutputDirPath](#)
- [SetPCBOutlineHeight](#)
- [SetPCBOutlineSettings](#)
- [SetPCBOutlineWidth](#)
- [SetPinNameASNetName](#)
- [SetPowerRegulator](#)
- [SetPowerRegulatorVoltage](#)
- [SetPreservePins](#)
- [SetPreserveUnusedPinsInBank](#)
- [SetRulesWorkingDir](#)
- [SetSchematicBoardFileName](#)
- [SetSchematicBoardFilePath](#)
- [SetSearchNReplaceNetNamePattern](#)
- [SetSkipConnectedPins](#)
- [SetSnapShotTimeInterval](#)
- [SetSymbolPinDirection](#)
- [SetUseBanks](#)
- [SetUseBanksForMultiDevices](#)
- [SetUseBanksForProtocol](#)
- [SetVoltageForMultiVoltagePins](#)

- [SpecifyDiffPinTermination](#)
- [SpecifyNetNames](#)
- [SpecifyOtherEndTermination](#)
- [SplitSymbolBankWise](#)
- [SplitSymbolConnectionWise](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)
- [StringTest](#)
- [SwapGroups](#)
- [TargetDevice](#)
- [TargetToMultipleDevices](#)
- [ToggleOptimizeTDConnectorUtilization](#)
- [ToggleSecondPassOptimization](#)
- [UnLockInstanceNets](#)
- [UnlockNets](#)
- [UnPreservePairPins](#)
- [UnPreserveTrueDifferentialPins](#)
- [UpdateAllegroSchematics](#)
- [UpdateAssignedToPinsWithConnectedPins](#)
- [UpdateDeepNWideGroupName](#)
- [UpdateDesignFromAllegro](#)
- [UpdateDeviceDataBase](#)
- [UpdateFPGAPortsFromCSV](#)
- [UpdateInstanceFootprint](#)
- [UpdateInstanceLocation](#)
- [UpdateInstancePartFromCSV](#)
- [UpdateInstanceSymbolFromCSV](#)
- [UpdateLayoutData](#)
- [UpdateNetGroup](#)
- [UpdateOrCADSchematics](#)
- [UpdatePartDescription](#)
- [UpdatePartFromCSV](#)
- [UpdateProtocolFromCSV](#)
- [UpdateVIFromCSV](#)
- [UpRevDesignDatabase](#)
- [UpRevLibraryPartDatabase](#)
- [Version](#)
- [ZoomFitAll](#)
- [ZoomFitInstance](#)

AddCustomMenuCommand

Adds the specified command to the Custom Menu.

Return

void

Syntax

```
AddCustomMenuCommand menu_name script_path_name
```

Parameters

Parameter	Description	Type	Optional
menu_name	String that appears in the Custom drop-down menu.	string	false
script_path_name	Specifies the path to the TCL script directory.	string	false

Examples

```
AddCustomMenuCommand \"Archive Design\" \"%%CDSROOT%/tools/fsp/scripts/archive_design.tcl\"
```

Related Commands

[AddCustomMenuCommand](#)

AddDecap

Assigns the specified decap to the specified power regulators.

Return

bool

Syntax

```
AddDecap -i instance_name -r power_regulator -g low_power_regulator -sp port_name -gp port_name -s schematic_symbol_info [-c decap_count]
```

Parameters

Parameter	Description	Type	Optional
-i	Specifies the name of the instance on which the decaps is to be assigned.	string	false
-s	Specifies the schematic symbol information (string returned by Component Browser).	string	false
-r	Specifies the name of the Supply(High) power regulator.	string	false
-g	Specifies the name of the Ground(Low) power regulator.	string	false
-sp	Specifies the name of the port that exist in the Supply side of the regulator.	string	false
-gp	Specifies the name of the port that exist in the Ground side of the regulator.	string	false
-c	Specifies the number of decaps that is to be added to the specified regulator. Default value is 1.	int	true

Examples

```
AddDecap -i U1 -s \"add <classlib>cap add :%Value:PACK_TYPE=SMDCAP :%DONTANOTATE:JEDEC_TYPE= :%DONTANOTATE:PART_NAME=CAP  
<classlib>cap.sym_1\" -r V_2_5 -g GND -sp A -gp B -c 10
```

Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)
- [GetDesignDecapLibrarySymbolNameList](#)

AddDecapOrCAD

Assigns decaps to the specified power regulators in the OrCAD schematics design.

Return

bool

Syntax

```
AddDecapOrCAD -i instance_name -r power_regulator -g low_power_regulator -sp port_name -gp port_name -o olb_name -p package_name [-c decap_count]
```

Parameters

Parameter	Description	Type	Optional
-i	Specifies the name of the instance on which the decaps is to be assigned.	string	false
-o	Specifies the name of the OrCAD symbol library file.	string	false
-p	Specifies the name of the OrCAD package.	string	false
-r	Specifies the name of the Supply(High) power regulator.	string	false
-g	Specifies the name of the Ground(Low) power regulator.	string	false
-sp	Specifies the name of the port that exist in the Supply side of the regulator.	string	false
-gp	Specifies the name of the port that exist in the Ground side of the regulator.	string	false
-c	Specifies the number of decaps to be assigned to the specified regulator. Default value is 1.	int	true

Examples

```
AddDecapOrCAD -i U6 -o %cdsroot%/tools/capture/library/Discrete.olb -p CAP -g GND -r V_1_5 -sp 1 -gp 2
```

Related Commands

- [AddDecap](#)
- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTerminationOrCAD](#)
- [MapTermination](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)
- [GetDesignDecapLibrarySymbolNameList](#)
- [ChangeGlobalOrCADLibraryPath](#)

AddInstanceCustomAttribute

Adds a custom attribute to the specified instance.

Return

bool

Syntax

```
AddInstanceCustomAttribute instance_name key value
```

Parameters

Parameter	Description	Type	Optional
instance_name	Specifies the name of the instance to which you want to add a custom attribute.	string	false
key	Specifies the name of the key that is to be added to the specified instance.	string	false
value	Specifies the value of the key.	string	false

Examples

```
AddInstanceCustomAttribute U1 part_description \"FPGA with 1152 pins\"
```

Related Commands

- [DeleteAllInstanceCustomAttributes](#)
- [DeleteInstanceCustomAttribute](#)
- [GetInstanceCustomAttributeValue](#)

AddInstancePinCustomAttribute

Adds custom attribute to the pin of the specified instance.

Return

bool

Syntax

```
AddInstancePinCustomAttribute instance_name pinNumber key value
```

Parameters

Parameter	Description	Type	Optional
instance_name	Specifies the name of the instance.	string	false
pinNumber	Specifies the name of the instance pin to which you want to add custom attribute.	string	false
key	Specifies the name of the key that is to be added to the selected instance pin.	string	false
value	Specifies the value of the specified key.	string	false

Examples

```
AddInstancePinCustomAttribute U2 G18 pin_description \"bank_1_pin\"
```

Related Commands

- [DeleteInstancePinCustomAttribute](#)
- [GetInstancePinCustomAttributeValue](#)

AddNet

Captures the connectivity between the specified instances. The net name is applied based on the template defined in the Settings form.

Return

bool

Syntax

```
AddNet sourceInstName destInstName sourcePinNumber destPinNumber
```

Parameters

Parameter	Description	Type	Optional
sourceInstName	Name of the interface instance to which the net is to be added.	string	false
destInstName	Name of the FPGA instance to which the net is to be added.	string	false
sourcePinNumber	Pin number of the interface instance.	string	false
destPinNumber	Pin number of the FPGA instance.	string	false

Examples

- `AddNet XP2 U2 188 AD21`
- `AddNet XP2 U2 120 H1`

Related Commands

- [DeleteNets](#)
- [ClearNets](#)
- [MovePinNet](#)
- [MoveDeviceNet](#)

AddNetCustomAttribute

Adds a custom attribute to the specified net.

Return

bool

Syntax

```
AddNetCustomAttribute net_name key value
```

Parameters

Parameter	Description	Type	Optional
net_name	Specifies the name of the net on which the custom attribute is to be added.	string	false
key	Specifies the name of the key that is to be added to the specified net.	string	false
value	Specifies the value of the specifies key.	string	false

Examples

```
AddInstanceCustomAttribute U1 part_description \"FPGA with 1152 pins\"
```

Related Commands

- [DeleteNetCustomAttribute](#)

AddPart

Places the specified schematic symbol part in the Canvas.

Return

string

Syntax

```
AddPart -s schematic_symbolinfo [-f family_name] [-g generate_rules_mapping_files)] [-lmf mapping_file] [-lrf rules_file] [-u] [-xloc X_location] [-yloc Y_location]
```

Parameters

Parameter	Description	Type	Optional
-s	Specifies the information of the schematic symbol. This string is returned from the Component Browser.	string	false
-u	Specifies whether to use the specified rules and mapping definition or generate it. In case this argument is not specified, the command will generate new rules and mapping file before creating instance.	bool	true
-lrf	Specifies the name of the rules file that is to be mapped with the specified symbol.	string	true
-lmf	Specifies the name of the mapping file that contains the mapping details of the rules file and symbol file.	string	true
-g	Specifies the option to generate the definition for the rules and mapping files. Use value as '\i\' to generate interface part, '\c\' to generate connector part, '\t\' to generate tester connector part. Default value is '\i\'.	string	true
-f	Specifies the comma separated target family names to be used for generated rules file. Default value is NONE.	string	true
-xloc	Specifies the center X location where the part is to be placed.	double	true
-yloc	Specifies the center Y location where the part is to be placed.	double	true

Examples

```
AddPart add <classlib>ls00 \"add :%Value:PACK_TYPE=DIP :%DONTANOTATE:JEDEC_TYPE=DIP14_3 :%DONTANOTATE:PART_NAME=74LS00  
<classlib>ls00.sym_1\"
```

Related Commands

- [PlaceInstance](#)
- [GetSupportedFamilyNames](#)
- [GetTargetFPGAFamilies](#)
- [GetDeviceFamilyName](#)
- [AddPartOrCAD](#)
- [AddPartModel](#)

AddPartModel

Creates a new rules file by importing the existing rules file.

Return

bool

Syntax

```
AddPartModel existinglrfPath newlrfPath targetDeviceFamily
```

Parameters

Parameter	Description	Type	Optional
existingLRFFFilePath	Specifies the path and name of the existing rules name to be imported.	string	false
newLRFFFilePath	Specifies the path and name of the rules file to be saved.	string	false
familyToTarget	Specifies the device family name to which the new rules file is to be targeted.	string	false

Examples

```
AddPartModel ddr2_sdram_x16_sd_84bga_v7.lrf ddr2_sdram_x16_sd_84bga_k7.lrf Kintex7
```

Related Commands

- [GetSupportedFamilyNames](#)
- [UpdatePartDescription](#)
- [GetDeviceFamilyName](#)
- [GetTargetFPGAFamilies](#)
- [AddPart](#)
- [AddPartOrCAD](#)
- [PlaceInstance](#)

AddPartOrCAD

Places the specified OrCAD symbol part in the Canvas.

Return

string

Syntax

```
AddPartOrCAD -o library_name -p package_name -j footprint_name [-f {family_names}] [-g generate_rules_and_mapping_files] [-lmf mapping_file] [-lrf rules_file] [-u use_existing_rules_and_mapping_files] [-xloc x_location] [-yloc y_location]
```

Parameters

Parameter	Description	Type	Optional
-o	Specifies the name of the OrCAD symbol library file that need to placed in the Canvas.	string	false
-p	Specifies the name of the OrCAD package of the specifies OrCAD symbol.	string	false
-j	Specifies the name of the footprint that is attached to the specified package.	string	false
-u	Specifies whether to use the specified rules and mapping definition or generate it. In case this argument is not specified, the command will generate new rules and mapping file before creating instance.	bool	true
-lrf	Specifies the name of the rules file that need to be mapped to the specified OrCAD symbol.	string	true
-lmf	Specifies the name of the mapping file that contains the mapping details of the rules file and OrCAD symbol.	string	true
-g	Specifies the option to generate the definition for the rules and mapping files. Use value as '\i' to generate interface part, '\c' to generate connector part, '\t' to generate tester connector part. Default value is '\i'.	string	true
-f	Specifies the comma separated target family names to be used for generated rules file. Default value is NONE.	string	true
-xloc	Specifies the center X location where the part is to be placed.	double	true
-yloc	Specifies the center Y location where the part is to be placed.	double	true

Examples

- `AddPartOrCAD -o C:/SPB_Data/fsp_working/test_interns_orcad/output/OrCAD/FSP_FE_LIB.OLB -p test_part -j cy7c1418av18`
- `AddPartOrCAD -o C:/SPB_Data/fsp_working/test_interns_orcad/output/OrCAD/FSP_FE_LIB.OLB -p test_part -j cy7c1418av18 -xloc 5 -yloc 5 -g t`
- `AddPartOrCAD -o %cdsroot%/tools/fsp/samples/orcad/cypress/cypress.olb -p cy7c1321av18 -j cy7c1321av18 -u -lrf cy7c1321av18 -lmf cy7c1321av18 -f {V4}`

Related Commands

- [PlaceInstance](#)
- [GetSupportedFamilyNames](#)
- [GetTargetFPGAFamilies](#)
- [GetDeviceFamilyName](#)
- [AddPart](#)
- [AddPartModel](#)
- [ChangeGlobalOrCADLibraryPath](#)

AddPowerRegulator

Adds power regulators to the design.

Return

bool

Syntax

```
AddPowerRegulator regulatorName voltage
```

Parameters

Parameter	Description	Type	Optional
regulatorName	Name of the power regulator that is to be added.	string	false
voltage	Specifies the supply voltage value for the specified regulator.	float	false

Examples

```
AddPowerRegulator V_0_9 0.9
```

Related Commands

- [AutoAddPowerRegulators](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

AddRulesFilePath

Adds a directory path to the rules search path where the rules files located.

Return

void

Syntax

```
AddRulesFilePath rules_dir_path
```

Parameters

Parameter	Description	Type	Optional
rules_dir_path	Specifies the directory path where the rules files are located.	string	false

Examples

```
AddRulesFilePath %cdsroot%/tools/fsp/samples/lrf
```

Related Commands

[GetRulesFilePaths](#)

AddSearchNReplaceNetNamePattern

Replaces the existing pattern of the net names with the specified pattern in the design.

Return

bool

Parameters

Parameter	Description	Type	Optional
searchString	Specifies the pattern to be replaced.	string	false
replaceString	Specifies the pattern to be used instead of the replaced pattern.	string	false

Examples

No Examples

Related Commands

- [GetSearchNReplaceNetNamePattern](#)
- [SetSearchNReplaceNetNamePattern](#)
- [RemoveSearchNReplaceNetNamePatterns](#)

AddSecondary

Adds the specified interface as a secondary interface to the specified Deep and Wide common group.

Return

bool

Syntax

```
AddSecondary common_group_name secondary_interface_instance_name {secondary_pinname_list}
```

Parameters

Parameter	Description	Type	Optional
common_group_name	Specifies the existing Deep and Wide common group name to which you want to add the specified interface instance.	string	false
secondary_interface_instance_name	Specifies the name of the interface instance that is to be added as secondary interface to the specified deep and wide group.	string	false
secondary_pinname_list	Specifies a list of pin names of the specified interface that is to be included in the specified Deep and Wide common group.	string_list	false

Examples

```
AddSecondary CommonGroup4 U4 {BA<0> BA<1> BA<2>}
```

Related Commands

- [CreateCommonGroup](#)
- [GetDeepNWideGroups](#)
- [RemoveDeepNWideGroup](#)
- [UpdateDeepNWideGroupName](#)
- [GetDeepNWideGroupInstanceList](#)
- [CreateWideBus](#)

AddTermination

Adds the specified termination in the current design.

Return

bool

Syntax

```
AddTermination -n termination_name -t termination_type
```

Parameters

Parameter	Description	Type
-n	Specifies the name of the termination that is to be added in the current design.	string
-t	Specifies the valid string to add the termination type. For example, \series\, \differential\, \differentialseriesparallel\, \pullupdown\, \split\, \powerfilter\, \isolvpc\, \parallel\, \thevenin\, \pfthevenin\, \pfhookup\.	string

Examples

- `AddTermination -n power_filter -t \"powerfilter\"`
- `AddTermination -n ser_term -t series`
- `AddTermination -n thev_term -t thevenin`

Related Commands

- [MapTerminationOrCAD](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)

AllocatePairPinsTogether

Specifies whether to use the device pair pins together while connecting single ended interface signals.

Return

void

Syntax

```
AllocatePairPinsTogether device_instance_name value_to_allocate_or_not
```

Parameters

Parameter	Description	Type	Optional
device_instance_name	Specifies the name of the device instance.	string	false
value_to_allocate_or_not	Specifies the whether to allocate the device pair pins together. The default value is true.	bool	false

Examples

- `AllocatePairPinsTogether U1 true`
- `AllocatePairPinsTogether U2 false`

Related Commands

[GetDeviceInstanceList](#)

ArchiveProject

Archives active FSP (DE-HDL only) project. This command archives used schematic symbol, rules, mapping file and dra. This command is not applicable in OrCAD schematic environment.

Return

bool

Syntax

```
ArchiveProject
```

Parameters

Parameter	Description	Type	Optional
-----------	-------------	------	----------

Examples

```
ArchiveProject
```

Related Commands

[OpenDEHDLProject](#)

AutoAddPowerRegulators

Defines a set of power regulators with voltage values as per the logical data of the instances and its connectivity.

Return

bool

Syntax

```
AutoAddPowerRegulators
```

Parameters

Parameter	Description	Type	Optional
-----------	-------------	------	----------

Examples

```
AutoAddPowerRegulators
```

Related Commands

- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

AutoMapPowerRegulators

Assigns the defined power regulators to the power pins of the instances. Ensure that required power regulators are available in design before using this command.

Return

bool

Syntax

```
AutoMapPowerRegulators [-preserve_power_regulator_mapping] [-preserve_instance_regulator_settings]
```

Parameters

Parameter	Description	Type	Optional
<code>-preserve_power_regulator_mapping</code>	Use this option to retain existing power regulator mapping and map regulator only for pins which are not connected to power regulator. By default, resets existing power mapping and connects power regulator as per voltage requirement of given pin.	bool	true
<code>-preserve_instance_regulator_settings</code>	Use this option to keep power regulator preference specified for each instance. Ensure that regulator preference is specified properly against each instance in 'Power Connection' window before using this option. By default, command overrides it by using all regulators for all instances.	bool	true

Examples

- `AutoMapPowerRegulators`
- `AutoMapPowerRegulators -preserve_power_regulator_mapping`
- `AutoMapPowerRegulators -preserve_instance_regulator_settings`
- `AutoMapPowerRegulators -preserve_power_regulator_mapping -preserve_instance_regulator_settings`

Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

AutoNetGroupDesignGroupWise

Creates NetGroups on all the interfaces, protocols, and virtual interfaces based on their logical groups.

Return

bool

Syntax

```
AutoNetGroupDesignGroupWise
```

Parameters

Parameter	Description	Type	Optional
-----------	-------------	------	----------

Examples

```
AutoNetGroupDesignGroupWise
```

Related Commands

- [AutoNetGroupInterface](#)
- [AutoNetGroupGroupWise](#)
- [AutoNetGroupSwappablePins](#)

AutoNetGroupDUTBasedOnConnectedDevices

This command is targeted to tester boards. To use BBO, each NetGroup should consist of nets connecting DUT to at most 1 connector. For tester boards having DUTs targeted to multiple connectors, this command auto-creates NetGroups based on the design connectivity to facilitate using BBO.

Return

bool

Syntax

```
AutoNetGroupDUTBasedOnConnectedDevices -instance dut_instance_name [-groups {group_name_list} ] [-prefix_dut_instance_name] [-prefix_group_name]
```

Parameters

Parameter	Description	Type	Optional
-instance	Name of the DUT instance.	string	false
-groups	Name of DUT groups. In case not specified, FSP will create netgroups for all groups of DUT.	string_list	true
-prefix_dut_instance_name	Adds DUT instance name as prefix to the generated netgroup name. If option is not specified, command will not add any instance name prefix to newly generated netgroup name.	bool	true
-prefix_group_name	Adds group name as prefix to the generated netgroup name. If option is not specified, command will not add any group name prefix to newly generated netgroup name.	bool	true

Examples

- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -prefix_dut_instance_name`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -prefix_dut_instance_name -prefix_group_name`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -groups [list group1 group2 group3]`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -groups [list group1:5]`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -groups [list group*] - prefix_dut_instance_name`
- `AutoNetGroupDUTBasedOnConnectedDevices -instance DUT -groups [list group*] - prefix_dut_instance_name -prefix_group_name`

Related Commands

- [AutoNetGroupDesignGroupWise](#)
- [AutoNetGroupInterface](#)
- [AutoNetGroupGroupWise](#)
- [AutoNetGroupSwappablePins](#)

AutoNetGroupGroupWise

This command automatically creates the NetGroups on the specified interface/protocol/virtual interface based on the logical groups.

Return

bool

Syntax

```
AutoNetGroupGroupWise -i interface_or_protocol_or_vi_name [-g group_name_list]
```

Parameters

Parameter	Description	Type	Optional
-i	Specifies the name of the interface, protocol, or virtual interface for which the NetGroups have to be created.	string	false
-g	Specifies the list of group names. If this argument is not specified, then net grouping will be done on the entire interface, protocol, or virtual interface.	string_list	true

Examples

- `AutoNetGroupGroupWise -i XP1`
- `AutoNetGroupGroupWise -i XP1 -g [list Address_control Data_Input]`

Related Commands

- [AutoNetGroupInterface](#)
- [AutoNetGroupSwappablePins](#)

AutoNetGroupInterface

This command automatically creates NetGroups based on the interface/protocol/virtual interface, interface or protocol wise.

Return

bool

Syntax

```
AutoNetGroupInterface -i interface_or_protocol_or_vi_name
```

Parameters

Parameter	Description	Type	Optional
-i	Specifies the name of the Interface/Protocol/Virtual Interface for which the NetGroup is to be created.	string	false

Examples

```
AutoNetGroupInterface -i XP1
```

Related Commands

- [AutoNetGroupGroupWise](#)
- [AutoNetGroupSwappablePins](#)

AutoNetGroupSwappablePins

Creates the NetGroups for the specified interface/protocol/virtual interface based on the swappable pins.

Return

bool

Syntax

```
AutoNetGroupSwappablePins -i interface_or_protocol_or_vi_name [-g group_name_list]
```

Parameters

Parameter	Description	Type	Optional
-i	Specifies the name of the interface or protocol, or virtual interface for which the NetGroups have to be created.	string	false
-g	Specifies the list of group names. If this argument is not specified, the NetGroups will be defined on the entire interface, protocol or virtual interface.	string_list	true

Examples

- `AutoNetGroupSwappablePins -i XP1`
- `AutoNetGroupSwappablePins -i XP1 -g [list Address_control Data_Input]`

Related Commands

- [AutoNetGroupInterface](#)
- [AutoNetGroupSwappablePins](#)

AutoSplitSymbol

Splits the symbol based on the number of pins allowed per symbol.

Return

bool

Syntax

```
AutoSplitSymbol instance_name maxPinsPerSplit
```

Parameters

Parameter	Description	Type	Optional
instance_name	Specifies the name of the instance for which symbol is to be split.	string	false
maxPinsPerSplit	Specifies the maximum number of pins that is to be allowed per symbol.	int	false

Examples

```
AutoSplitSymbol U2 100
```

Related Commands

- [SplitSymbolConnectionWise](#)
- [SplitSymbolBankWise](#)
- [MergeAllSymbolSplits](#)

AutoUpdateAllDeviceFPGAPortNames

Updates the FPGA port names for all the connected pins with the net names in the design.

Return

bool

Syntax

```
AutoUpdateAllDeviceFPGAPortNames
```

Parameters

No Parameters

Examples

```
AutoUpdateAllDeviceFPGAPortNames
```

Related Commands

[AutoUpdateDeviceFPGAPortNames](#)

AutoUpdateDeviceFPGAPortNames

Updates the FPGA port names of all the signals that are connected to the specified FPGA based on connected net name. Please note that port names will not be updated for the un-routed signals or signals with the port names.

Return

bool

Syntax

```
AutoUpdateDeviceFPGAPortNames [-device_name name] [-override]
```

Parameters

Parameter	Description	Type	Optional
-device_name	Specifies the name of the device instance for which the FPGA port names is to be updated. If not specified, command automatically updates port names for all FPGA.	string	true
-override	Specifies if the FPGA port names is to be always updated as per connected net name. By default, command retains already defined port names.	bool	true

Examples

- `AutoUpdateDeviceFPGAPortNames`
- `AutoUpdateDeviceFPGAPortNames -override`
- `AutoUpdateDeviceFPGAPortNames -device_name U1`
- `AutoUpdateDeviceFPGAPortNames -device_name U5 -override`

Related Commands

[ResetMapppedPorts](#)

BoolTest

Tests the bool variable type.

Return

bool

Syntax

```
BoolTest arg
```

Parameters

Parameter	Description	Type	Optional
arg	Specifies the variable type value that is to be tested.	bool	false

Examples

- `BoolTest true`
- `BoolTest 1`

Related Commands

- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

ChangeGlobalOrCADLibraryPath

Changes OrCAD library path reference in FSP database. OrCAD library paths can be referenced by instances, termniations, decaps etc.. This command is applicable in OrCAD schematic environment.

Return

bool

Syntax

```
ChangeGlobalOrCADLibraryPath [existing_olb_file_path new_olb_file_path] [-report]
```

Parameters

Parameter	Description	Type	Optional
existing_olb_file_path	File path of OrCAD schematic symbol library used in active design.	string	true
new_olb_file_path	File path of OrCAD schematic symbol library to be replaced by in design.	string	true
-report	Reports all path of OrCAD schematic symbol library used in design.	bool	true

Examples

- `ChangeGlobalOrCADLibraryPath -report`
- `ChangeGlobalOrCADLibraryPath \"./orcad_extern/output/OrCAD/FSP_FE_LIB.OLB\"
\"F:/designs_data/fsp_designs/orcad_extern/output/OrCAD/FSP_FE_LIB.OLB\"`
- `ChangeGlobalOrCADLibraryPath \"./orcad_extern/output/OrCAD/FSP_FE_LIB.OLB\" \"$MY_ORCAD_LIBS/sample_lib.OLB\"`
- `ChangeGlobalOrCADLibraryPath \"$MY_ORCAD_LIBS/sample_lib.OLB\" \"$CDS_SITE/all_orcad_libs/sample_lib.OLB\"`

Related Commands

- [LinkToFESymbol](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)

ChangeNetName

Changes the existing net name to the specified new name.

Return

bool

Syntax

```
ChangeNetName presentNetName newNetName
```

Parameters

Parameter	Description	Type	Optional
presentNetName	Specifies the name of the net in the design.	string	false
newNetName	Specifies the name of the net that is to be assigned to the selected net.	string	false

Examples

```
ChangeNetName XP1_DDR_A <6> my_net
```

Related Commands

- [GetNetName](#)

ChangeProtocolOrder

Changes the device instance order for the specified protocol.

Return

bool

Syntax

```
ChangeProtocolOrder protocol_name new_ordered_device_name_list
```

Parameters

Parameter	Description	Type	Optional
protocol_name	Specifies the name of the protocol of which the device instances order need to be changed.	string	false
new_ordered_device_name_list	Specifies the list of the device instance names in a required order.	stringstring_list	false

Examples

- ```
ChangeProtocolOrder U1_U2_U3_U4 [list U4 U3 U2 U1]
```

## Related Commands

- [CreateProtocolFromCSV](#)
- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromLibraryModel](#)

## ChangeRules

Changes the link of the rules file for the specified instance. This command is useful when the instance is linked to a schematic symbol and required to switch from one rules definition to another.

### Return

bool

### Syntax

```
ChangeRules -instance instance_name -rules rulesFile [-mapping mappingFile]
```

## Parameters

| Parameter | Description                                                                        | Type   | Optional |
|-----------|------------------------------------------------------------------------------------|--------|----------|
| -instance | Specifies the name of the instance for which the rules file link is to be changed. | string | false    |
| -rules    | Specifies the name of the rules file that is to be linked.                         | string | false    |
| -mapping  | Specifies the name of the existing mapping file.                                   | string | true     |

## Examples

```
ChangeRules -instance U21 -rules mt46v64m4__v4__v5 -mapping mt46v64m4
```

## Related Commands

[GetInstanceNameList](#)

# ChangeSchematicsSymbolFileReference

Changes library file(.olb) reference for the instance(OrCAD design).

## Return

bool

## Syntax

```
ChangeSchematicsSymbolFileReference -i instance_naem -o orcad_library_name
```

## Parameters

| Parameter | Description                                   | Type   | Optional |
|-----------|-----------------------------------------------|--------|----------|
| -i        | Specifies the name of the instance.           | string | false    |
| -o        | Specifies the name of the Orcad library file. | string | false    |

## Examples

```
ChangeSchematicsSymbolFileReference - U1 -o d:/test.olb
```

## Related Commands

[GetSchematicsSymbolFileReference](#)

# CheckDesignConsistency

Checks consistency between the rules, mapping, schematic symbol,and dra files. After running this command, is any files or data that are either missing or mismatching with the central library database will be reported.

## Return

string\_list

## Syntax

```
CheckDesignConsistency -update {instance_list}
```

## Parameters

| Parameter | Description                                             | Type        | Optional |
|-----------|---------------------------------------------------------|-------------|----------|
| -update   | Specifies the parts to be sync with library definition. | string_list | true     |

## Examples

- `CheckDesignConsistency`
- `CheckDesignConsistency -update {U1 I1 U4}`

## Related Commands

- [ReportDesignFileReferences](#)
- [ReportAllDRAFiles](#)
- [ReportAllMappingFiles](#)
- [ReportAllRulesFiles](#)
- [ReportAllFPGAPinMappingFiles](#)

# ClearAllMessageWindows

Clears log message texts from Log window, warning texts from Warning window,and error message texts from the Error window.

## Return

void

## Syntax

```
ClearAllMessageWindows
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ClearAllMessageWindows
```

## Related Commands

[ClearAllMessageWindows](#)

# ClearNets

Removes the connectivity/nets for the open design.

## Return

bool

## Syntax

ClearNets

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

ClearNets

## Related Commands

- [DeleteNets](#)

# CloseProject

Closes the current project.

## Return

bool

## Syntax

CloseProject

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

CloseProject

## Related Commands

- [OpenProject](#)
- [SaveProject](#)
- [SaveProjectAs](#)

# ConvertLRFTToRealPart

Converts the specified rules file to a real interface instance.

## Return

bool

## Syntax

```
ConvertLRFTToRealPart -i instance_name -lmf mapping_file -s schematic_symbol_info
```

## Parameters

| Parameter | Description                                                                        | Type   | Optional |
|-----------|------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of instance that is to be converted to a real part.             | string | false    |
| -lmf      | Specifies the path to the mapping file that is to be used for mapping.             | string | false    |
| -s        | Specifies the schematic symbol information (string returned by Component Browser). | string | false    |

## Examples

```
ConvertLRFTToRealPart -i XPl -lmf ddr2_dimm_x4 -s \"add <connectors>ddr2_dimm_x4 add :%Value:PART_NAME=DDR2_DIMM_X4
:%DONTANOTATE:FSP_LOGICAL_MODEL=ddr2_dimm_x4__a2gx,ddr2_dimm_x4__sp3a,ddr2_dimm_x4__sp3,ddr2_dimm_x4__s3__s4gx__s4e,ddr2_dimm_x4__s2__s2gx__
agx,ddr2_dimm_x4__v4__v5 <connectors>ddr2_dimm_x4.sym_1\"
```

## Related Commands

- [LinkToFESymbol](#)
- [LinkToFESymbolOrCAD](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)

## ConvertLRFTToRealPartOrCAD

Converts the interface to a real interface by linking with schematic symbol and footprint. The command uses the same schematic symbol while generating schematics.

## Return

bool

## Syntax

```
ConvertLRFTToRealPartOrCAD -i instance_name -lmf mapping_file -o complete_olb_file_path -p package_name -j jedec_type
```

## Parameters

| Parameter | Description                                                              | Type   | Optional |
|-----------|--------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of instance that is to be converted to a real part.   | string | false    |
| -lmf      | Specifies the path to the mapping file that is to be used for mapping.   | string | false    |
| -o        | Specifies the path to the OrCAD schematic symbol library.                | string | false    |
| -p        | Specifies the name of the package for the specified schematic symbol.    | string | false    |
| -j        | Specifies the name of the Jedec type for the specified schematic symbol. | string | false    |

## Examples

```
ConvertLRFTToRealPartOrCAD -i U4 -lmf cy7c1315bv18 -o \"%cdsroot%/tools/fsp/samples/orcad/cypress/cypress.olb\" -p cy7c1315bv18 -j cy7c1315bv18
```

## Related Commands

- [LinkToFESymbol](#)
- [LinkToFESymbolOrCAD](#)
- [ConvertLRFTToRealPart](#)
- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)
- [ChangeGlobalOrCADLibraryPath](#)

## ConvertVIToRealInterface

Converts the specified virtual interface instance to a real interface instance.

## Return

bool

## Syntax

```
ConvertVIToRealInterface -i instance_name -lrf rules_file -lmf mapping_file -s schematic_symbol_info
```

## Parameters

| Parameter | Description                                                                                                                                                               | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the virtual interface that is to be converted to a real interface.                                                                                  | string | false    |
| -lrf      | Specifies the name of the rules file to be generated for the specified virtual interface.                                                                                 | string | false    |
| -lmf      | Specifies the path to the mapping file that need to be used for mapping. The mapping file contains the mapping details of the virtual interface and the schematic symbol. | string | false    |
| -s        | Specifies the schematic symbol information (string returned by Component Browser).                                                                                        | string | false    |

## Examples

```
ConvertVIToRealInterface -i U3_VI8 -lrf ddr2_dimm_x4 -lmf ddr2_dimm_x4 -s \"add <connectors>ddr2_dimm_x4 add :%Value:PART_NAME=DDR2_DIMM_X4
:%DONTANOTATE:FSP_LOGICAL_MODEL=ddr2_dimm_x4__a2gx,ddr2_dimm_x4__sp3a,ddr2_dimm_x4__sp3,ddr2_dimm_x4__s3__s4gx__s4e,ddr2_dimm_x4__s2__s2gx__
agx,ddr2_dimm_x4__v4__v5 <connectors>ddr2_dimm_x4.sym_1\"
```

## Related Commands

- [LinkToFESymbol](#)
- [LinkToFESymbolOrCAD](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterfaceOrCAD](#)

# ConvertVIToRealInterfaceOrCAD

Converts the specified virtual interface instance to a real interface instance.

## Return

bool

## Syntax

```
ConvertVIToRealInterfaceOrCAD -i instance_name -lrf rules_file -lmf mapping_file -o complete_olb_file_path -p package_name -j jedec_type
```

## Parameters

| Parameter | Description                                                                                                                                                             | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the virtual interface that is to be converted to a real interface.                                                                                | string | false    |
| -lrf      | Specifies the name of the rules file that is to be generated for the specified virtual interface.                                                                       | string | false    |
| -lmf      | Specifies the path to the mapping file that is to be used for mapping. The mapping file contains the mapping details of the virtual interface and the schematic symbol. | string | false    |
| -o        | Specifies the path to the OrCAD schematic symbol library.                                                                                                               | string | false    |
| -p        | Specifies the package name of the specified schematic symbol.                                                                                                           | string | false    |
| -j        | Specifies the Jedec type of the specified schematic symbol.                                                                                                             | string | false    |

## Examples

```
ConvertVIToRealInterfaceOrCAD -i U3_VI12 -lrf cy7c1315bv18 -lmf cy7c1315bv18 -o \"%cdsroot%/tools/fsp/samples/orcad/cypress/cypress.olb\" -p
cy7c1315bv18 -j cy7c1315bv18
```



## Related Commands

- [LinkToFESymbol](#)
- [LinkToFESymbolOrCAD](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterface](#)
- [ChangeGlobalOrCADLibraryPath](#)

## CreateCommonGroup

Creates a Deep and Wide common group with the specified name. In addition, the specified interface name is set as primary interface. This command is expected to be followed by AddSecondary command using which you can add secondary interfaces to the Deep and Wide common group.

### Return

bool

### Syntax

```
CreateCommonGroup common_group_name primary_interface_name {common_pinname_list}
```

### Parameters

| Parameter              | Description                                                                                                               | Type        | Optional |
|------------------------|---------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| common_group_name      | Specifies the name that is to be assigned for the Deep and Wide common group.                                             | string      | false    |
| primary_interface_name | Specifies the name of the primary interface.                                                                              | string      | false    |
| common_pinname_list    | Specifies a list of pin names of the primary interface that should be a part of the specified Deep and Wide common group. | string_list | false    |

### Examples

```
CreateCommonGroup CommonGroup4 U3 {BA<0> BA<1> BA<2>}
```

## Related Commands

- [AddSecondary](#)
- [GetDeepNWideGroups](#)
- [RemoveDeepNWideGroup](#)
- [UpdateDeepNWideGroupName](#)
- [GetDeepNWideGroupInstanceList](#)

## CreateNewDEHDLProject

Creates a new DE-HDL schematics environment project in the specified path using the specified project name. As an optional argument you can specify design library and symbol library.

### Return

bool

## Syntax

```
CreateNewDEHDLProject [-design_path design_directory_path] [-name project_name] [-symbols_lib symbol_library_name] [-design_library design_library_name]
```

## Parameters

| Parameter       | Description                                                                                                                                                                                                    | Type   | Optional |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -name           | Specifies the name for the project that is to be created.                                                                                                                                                      | string | false    |
| -design_path    | Specifies the directory path in which the project has to be created.                                                                                                                                           | string | false    |
| -symbols_lib    | Specifies the name of the library in which the DE-HDL symbols is to be generated. If this argument is not specified, the fsp_fe_lib will be used as library by default.                                        | string | true     |
| -design_library | Specifies the design library name for the project. All the schematics block symbols will be generated in this library. If this argument is not specified, the project name will be used as library by default. | string | true     |

## Examples

- `CreateNewDEHDLProject -design_path C:/SPB_Data/fsp_working/tcl_test_setup -name tcl_command -symbols_lib tcl_command_lib -design_library design_library`
- `CreateNewDEHDLProject -design_path C:/SPB_Data/fsp_working/tcl_test_setup -name tcl_command`

## Related Commands

- [CreateNewOrCADProject](#)
- [OpenDEHDLProject](#)

## CreateNewNetGroup

Creates a new NetGroup for the specified pin numbers or signal names of the specified interface, protocol, or virtual interface.

## Return

bool

## Syntax

```
CreateNewNetGroup -i interface_or_protocol_or_vi_name -n net_group_name -p pin_number_or_port_name_list
```

## Parameters

| Parameter | Description                                                                                                  | Type        | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------|-------------|----------|
| -i        | Specifies the name of the interface, protocol, or virtual interface for which the NetGroup is to be created. | string      | false    |
| -n        | Specifies the name of the NetGroup to be created.                                                            | string      | false    |
| -p        | Specifies the list of pin numbers or signal names on which the NetGroup is to be created.                    | string_list | false    |

## Examples

```
CreateNewNetGroup -i XP1 -n XP1_NetGroup -p [list A1 A2 B1 B2]
```

## Related Commands

- [DeleteNetGroups](#)
- [UpdateNetGroup](#)

## CreateNewOrCADProject

Creates a new OrCAD schematics environment project in the specified path using the specified name.

## Return

bool

## Syntax

```
CreateNewOrCADProject -design_path directory_path -name project_name [-ini_file capture_ini_file]
```

## Parameters

| Parameter    | Description                                                                                                                            | Type   | Optional |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -name        | Specifies the name for the project to be created.                                                                                      | string | false    |
| -design_path | Specifies the directory path in which the FSP project is to be created.                                                                | string | false    |
| -ini_file    | Specifies the name of the OrCAD settings(.ini) file. If this argument is not specified, the command uses the default capture.ini file. | string | true     |

## Examples

- `CreateNewOrCADProject -dir C:/SPB_Data/fsp_working/tcl_test_setup -name tcl_command`
- `CreateNewOrCADProject -dir C:/SPB_Data/fsp_working/tcl_test_setup -name tcl_command -ini_file C:/SPB_Data/capture.ini`

## Related Commands

[CreateNewDEHDLProject](#)

## CreateNewProject

Creates a new project in the specified path using the specified name. This command also accepts project .cpm file name and symbol library name.

## Return

bool

## Syntax

```
CreateNewProject -dir directory_path -name project_name [-cpm cpm_file_Name [-lib symbol_library_name]
```

## Parameters

| Parameter | Description                                                                                                                                                                           | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -dir      | Specifies the directory path in which the project is to be created.                                                                                                                   | string | false    |
| -name     | Specifies the name for the project to be created.                                                                                                                                     | string | false    |
| -cpm      | Specifies the path and name of the .cpm file. In case this argument is not specified, the command automatically generates a cpm file for new project.                                 | string | true     |
| -lib      | Specifies the name of the library in which the DE-HDL symbols is to be generated. In case this argument is not specified, the command automatically uses the project name as library. | string | true     |

## Examples

```
CreateNewProject C:/SPB_Data/fsp_working/tcl_test_setup -name tcl_command -cpm C:/SPB_Data/fsp_working/tcl_test_setup/tcl_command.cpm -lib tcl_command_lib
```

## Related Commands

[NewProject](#)

# CreatePartFromCSV

Creates a part using the CSV file.

## Return

bool

## Syntax

```
CreatePartFromCSV -l rules_file_path -c csv_file_path -m column_mapping [-d delimiter] [-i ignore_rows] [-g is_connector]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                                             | Type              | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -l        | Specifies the name of the rules file.                                                                                                                                                                                                   | string            | false    |
| -c        | Specifies the path of the CSV file that is to be imported.                                                                                                                                                                              | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                                                                                             | string_string_map | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma),  (pipe), ;(semicolon), :(colon).                                                                                                                               | string            | true     |
| -i        | Specifies the row numbers in comma separated format, that are to be ignored during import from the CSV file. By default, this command does not ignore any rows.                                                                         | string            | true     |
| -g        | Specifies whether the generated rules file shall be of type connector or fixed pin interface. Use 'y' to generate connector rules file and 'n' to create interface rules file. By default, this command generates interface rules file. | string            | true     |

## Examples

- `CreatePartFromCSV -l create_part_csv_new.lrf -c ./create_part_csv.csv -m {"Group Name\" 3 \"Diff. Pair Pin\" 18 \"Pin Number\" 19 \"Target Pin Function\" 20 \"IO Standard\" 21 \"Pin Type\" 22 \"Diff. Type\" 23 \"Voltage Level\" 24 \"Pin Name\" 25} -d , -i 1`
- `CreatePartFromCSV -l create_part_csv_connector.lrf -c ./create_part_conn_csv.csv -m {"Group Name\" 3 \"Diff. Pair Pin\" 18 \"Pin Number\" 19 \"Target Pin Function\" 20 \"IO Standard\" 21 \"Pin Type\" 22 \"Diff. Type\" 23 \"Voltage Level\" 24 \"Pin Name\" 25} -d , -i 1 -g y`

## Related Commands

[UpdatePartFromCSV](#)

## CreateProtocolFromConstraintsPinoutfile

Creates a protocol from the specified pin outs and constraints file.

## Return

bool

## Syntax

```
CreateProtocolFromConstraintsPinoutfile {device_name_list} hdl_file_path pinout_file_path hdl_type
```

## Parameters

| Parameter        | Description                                                                               | Type        | Optional |
|------------------|-------------------------------------------------------------------------------------------|-------------|----------|
| device_name_list | Specifies the list of the devices instance names for which the protocol is to be created. | string_list | false    |
| hdl_file_path    | Specifies the hdl (verilog or vhd) file path.                                             | string      | false    |
| pinout_file_path | Specifies the constraint (ucf, xdc, qsf, fx, or pin) file path that is to be imported.    | string      | false    |
| hdl_type         | Specifies the module name of the hdl file.                                                | string      | false    |

## Examples

- `CreateProtocolFromConstraintsPinoutfile [list U1 U2] ./U1.v ./U1.ucf Verilog`
- `CreateProtocolFromConstraintsPinoutfile [list U3 U4] ./design.v ./design.qsf Verilog`
- `CreateProtocolFromConstraintsPinoutfile [list U5 U6] ./fpga.v ./fpga.pin Verilog`

## Related Commands

- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromLibraryModel](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromCSV](#)
- [ExportCSVFromProtocol](#)

## CreateProtocolFromCSV

Creates a protocol using the CSV file.

## Return

bool

## Syntax

```
CreateProtocolFromCSV -p {device_name_list} -c csv_file_path -m column_mapping [-d delimiter] [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -p        | Specifies the list of the devices instance names for which the protocol is to be created.                                                                      | string_list       | false    |
| -c        | Specifies the path of the csv file that is to be imported to create protocol.                                                                                  | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon).                                                         | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

```
CreateProtocolFromCSV -c ./create_protocol_csv.csv -d , -i 1 -p [list U1 U2] -m {"Diff. Pair Signal\" 5 \"NetGroup\" 6 \"Target Pin Function\" 7 \"IO Standard\" 8 \"Pin Type\" 9 \"Diff. Type\" 10 \"Port Name\" 11}
```

## Related Commands

- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromLibraryModel](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [ExportCSVFromProtocol](#)

## CreateProtocolFromExistingProtocol

Creates a device protocol by importing the protocol definition from an external file.

## Return

bool

## Syntax

```
CreateProtocolFromExistingProtocol {device_name_list} protocol_file_path
```

## Parameters

| Parameter          | Description                                                                               | Type        | Optional |
|--------------------|-------------------------------------------------------------------------------------------|-------------|----------|
| device_name_list   | Specifies the list of the devices instance names for which the protocol is to be created. | string_list | false    |
| protocol_file_path | Specifies the path and name of the external file that stores the protocol information.    | string      | false    |

## Examples

```
CreateProtocolFromExistingProtocol [list U1 U2] ./protocol.prf
```

## Related Commands

- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ExportProtocolDefinition](#)
- [CreateProtocolFromCSV](#)
- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [CreateProtocolFromLibraryModel](#)
- [RemoveProtocol](#)
- [RemoveAllProtocols](#)

## CreateProtocolFromLibraryModel

Creates a device protocol by importing the protocol definition from a rules file.

### Return

bool

### Syntax

```
CreateProtocolFromLibraryModel {device_name_list} rules_name
```

### Parameters

| Parameter        | Description                                                                               | Type        | Optional |
|------------------|-------------------------------------------------------------------------------------------|-------------|----------|
| device_name_list | Specifies the list of the devices instance names for which the protocol is to be created. | string_list | false    |
| rules_name       | Specifies the name of the rules file from which the pin definitions is to be imported.    | string      | false    |

## Examples

```
CreateProtocolFromLibraryModel [list U1 U2] ddr2_dimm
```

## Related Commands

- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ExportProtocolDefinition](#)
- [CreateProtocolFromCSV](#)
- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [CreateProtocolFromExistingProtocol](#)

## CreateSystemAceChain

Creates a system ace chain between the specified system interface and devices.

### Return

bool

### Syntax

```
CreateSystemAceChain systemace_interface_name {device_name_list}
```

### Parameters

| Parameter                | Description                                             | Type        | Optional |
|--------------------------|---------------------------------------------------------|-------------|----------|
| systemace_interface_name | Specifies the name of the SystemAce interface instance. | string      | false    |
| device_name_list         | Specifies a list of names of the device instances.      | string_list | false    |

### Examples

```
CreateSystemAceChain U7 {U8 U1 U5}
```

### Related Commands

[GetDeviceInstanceList](#)

## CreateTargetDeviceSet

Creates devices instance set inorder to connect multiple interafce pins to multiple devices.

### Return

bool

### Syntax

```
CreateTargetDeviceSet set_name {devices_names_list}
```

### Parameters

| Parameter          | Description                            | Type        | Optional |
|--------------------|----------------------------------------|-------------|----------|
| set_name           | Specifies the name of the set.         | string      | false    |
| devices_names_list | Specifies a list of device names list. | string_list | false    |



## Examples

- `CreateTargetDeviceSet mux_set {{U1 U2} {U4 U5 U6} {J1 J4 J6}}`
- `CreateTargetDeviceSet mux_set [list [list U1 U2] [list U4 U5 U6] [list J1 J4 J6]]`
- `CreateTargetDeviceSet MUX [list MUX1 MUX2 MUX3 MUX4 MUX5 MUX6]`
- `CreateTargetDeviceSet MUX [list MUX1:6]`
- `CreateTargetDeviceSet MUX [list MUX*]`

## Related Commands

[GetDeviceInstanceList](#)

## CreateVIFromCSV

Creates a new virtual interface by importing the CSV file.

## Return

string

## Syntax

```
CreateVIFromCSV -p device_name -c csv_file_path -m column_mapping [-d delimiter] [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -p        | Specifies the name of the device instance for which the virtual interface is to be created.                                                                    | string            | false    |
| -c        | Specifies the path of the csv file that is to be imported to create virtual interface.                                                                         | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon).                                                         | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

```
CreateVIFromCSV -c ./create_vi_csv.csv -d , -i 1 -m {"Diff. Pair Signal\" 5 \"Target Pin Function\" 6 \"IO Standard\" 7 \"Pin Type\" 8
\"Diff. Type\" 9 \"Port Name\" 12} -p U2
```

## Related Commands

- [UpdateVIFromCSV](#)
- [ExportCSVFromVI](#)
- [CreateVIFromExistingVI](#)
- [CreateVIFromLibraryModel](#)
- [ExportVirtualInterfaceDefinition](#)

## CreateVIFromExistingVI

Creates a virtual interface by importing the pin definition from an virtual interface definition file.

### Return

bool

### Syntax

```
CreateVIFromExistingVI deviceName existingViFile
```

### Parameters

| Parameter      | Description                                                                                                                      | Type   | Optional |
|----------------|----------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| deviceName     | Specifies the name of the device instance for which the virtual interface is to be created.                                      | string | false    |
| existingViFile | Specifies the name of the virtual interface file path from which the pin definitions of the virtual interface is to be imported. | string | false    |

### Examples

- `CreateVIFromExistingVI U4 [GetProjectFilePath]/vi.vrf`
- `CreateVIFromExistingVI U1 /export/home/fsp/projects/rules/memory_vi.vrf`

### Related Commands

- [ExportVirtualInterfaceDefinition](#)
- [CreateVIFromCSV](#)
- [UpdateVIFromCSV](#)
- [ExportCSVFromVI](#)

## CreateVIFromLibraryModel

Creates a virtual interface by importing the pin definition from the specified rules file.

### Return

bool

### Syntax

```
CreateVIFromLibraryModel deviceName libraryModel
```

### Parameters

| Parameter    | Description                                                                             | Type   | Optional |
|--------------|-----------------------------------------------------------------------------------------|--------|----------|
| deviceName   | Specifies the name of the device instance for which virtual interface is to be created. | string | false    |
| libraryModel | Specifies the name of the rules file from which the pin definition is to be imported.   | string | false    |

## Examples

```
CreateVIFFromLibraryModel U3 cy7c1418av18__v4__v5
```

## Related Commands

- [GetDeviceInstanceList](#)

## CreateVIFFromStringList

Creates a new virtual interface with given name, given device and given signal data. Sinals are specified in list format.

## Return

bool

## Syntax

```
CreateVIFFromStringList viName deviceInstanceName signalData
```

## Parameters

| Parameter          | Description                                                                                                                                  | Type                                                                 | Optional |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|----------|
| viName             | Specify virtual interface instance name you would like to use for newly created virtual interface                                            | string                                                               | false    |
| deviceInstanceName | string                                                                                                                                       | Device instance name for which virtual interface needs to be created | false    |
| signalData         | Specify signal data in list-list format where first list reperesents different signals while later one represents list of signal properties. | string_string_list                                                   | false    |

## Examples

```
CreateVIFFromStringList
```

## Related Commands

- [CreateVIFFromLibraryModel](#)
- [GetVIFInstanceList](#)
- [GetDeviceInstanceList](#)

## CreateVirtualInterfaceFromConstraintsPinoutfile

Creates a virtual interface from the specified pin outs and constraints file.

## Return

bool

## Syntax

```
CreateVirtualInterfaceFromConstraintsPinoutfile device_name hdl_file_path pinout_file_path hdl_type
```

## Parameters

| Parameter        | Description                                                                                 | Type   | Optional |
|------------------|---------------------------------------------------------------------------------------------|--------|----------|
| device_name      | Specifies the name of the device instance for which the virtual interface is to be created. | string | false    |
| hdl_file_path    | Specifies the path and name of the hdl (verilog or vhd) file that is to be imported.        | string | false    |
| pinout_file_path | Specifies the constraint (ucf, xdc, qsf, fx, or pin) file path that is to be imported.      | string | false    |
| hdl_type         | Specifies the module name of the hdl file.                                                  | string | false    |

## Examples

- `CreateVirtualInterfaceFromConstraintsPinoutfile U1 ./U1.v ./U1.ucf Verilog`
- `CreateVirtualInterfaceFromConstraintsPinoutfile U3 ./design.v ./design.qsf Verilog`
- `CreateVirtualInterfaceFromConstraintsPinoutfile U5 ./fpga.v ./fpga.pin Verilog`

## Related Commands

- [CreateVIFromCSV](#)
- [CreateVIFromExistingVI](#)
- [CreateVIFromLibraryModel](#)

# CreateVirtualInterfaceFromXDCfile

Creates a virtual interface from the specified pin outs and constraints file.

## Return

bool

## Syntax

```
CreateVirtualInterfaceFromXDCfile device_name xdc_file_path
```

## Parameters

| Parameter     | Description                                                                                 | Type   | Optional |
|---------------|---------------------------------------------------------------------------------------------|--------|----------|
| device_name   | Specifies the name of the device instance for which the virtual interface is to be created. | string | false    |
| xdc_file_path | Specifies path and name of the XDC file that is to be imported.                             | string | false    |

## Examples

```
CreateVirtualInterfaceFromXDCfile U1 ./U1.xdc
```

## Related Commands

- [CreateVIFromCSV](#)
- [CreateVIFromExistingVI](#)
- [CreateVIFromLibraryModel](#)

## CreateWideBus

Creates a wide bus with the specified name comprising of the interface signals or bus names in order.

### Return

bool

### Syntax

```
CreateWideBus wide_bus_name {interface_bus_name_list}
```

### Parameters

| Parameter               | Description                                                                                                                                                                                                                                | Type        | Optional |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| wide_bus_name           | Specifies the name of the bus.                                                                                                                                                                                                             | string      | false    |
| interface_bus_name_list | Specifies a list of interface signals names in order that should be the part of the specified wide bus. The signals/buses should be specified in the syntax {interface_name1.signal_or_bus_name1 interface_name2.signal_or_bus_name2 ...}. | string_list | false    |

### Examples

- `CreateWideBus my_bus {U15.A<0:4> U16.A<7:0>}`
- `CreateWideBus my_bus {U15.A<0:4> U15.A<9:6>}`

### Related Commands

- [CreateCommonGroup](#)
- [AddSecondary](#)
- [GetDeepNWideGroups](#)
- [RemoveDeepNWideGroup](#)
- [UpdateDeepNWideGroupName](#)
- [GetDeepNWideGroupInstanceList](#)

## DeferSpecialPurposePinsUsage

Specifies whether to reserve special purpose pins such as VREF, VRP, VRN, CLOCK while allocating pins in synthesis to the possible extent. By default, this option is set to true, which means during synthesis special purpose pins will be deferred for allocation to the possible extent.

### Return

void

### Syntax

```
DeferSpecialPurposePinsUsage device_instance_name value_to_differ_or_not
```

## Parameters

| Parameter              | Description                                      | Type   | Optional |
|------------------------|--------------------------------------------------|--------|----------|
| device_instance_name   | Specifies the name of the device instance.       | string | false    |
| value_to_differ_or_not | Specifies the value. The default value is false. | bool   | false    |

## Examples

- `DeferSpecialPurposePinsUsage U1 true`
- `DeferSpecialPurposePinsUsage U2 false`

## Related Commands

[GetDeviceInstanceList](#)

# DefineJTAGChain

Creates a JTAG chain in the current design.

## Return

bool

## Syntax

```
DefineJTAGChain [-name jtag_chain_name] -instanceList {list_of_instance_names} -inputNetNamesOfInstanceChain {tdo_tdi_pins_netnames_list} -
tckNetName tcknetname -tmsNetName tmsnetName [-loop loop_options]
```

## Parameters

| Parameter                         | Description                                                                                                                         | Type        | Optional |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -name                             | Specifies the name that is to be specified for the JTAG chain. If not specified, FSP automatically assign a name to the JTAG chain. | string      | true     |
| -loop                             | Specifies whether the JTAG chain is either open or close ended loop. Valid values are open or close. Default value is open.         | string      | true     |
| -instanceList                     | Specifies a list of instance names which requires to be connected in the JTAG chain.                                                | string_list | false    |
| -tckNetName                       | Specifies the net name that is to be assigned for the TCK pins.                                                                     | string      | false    |
| -tmsNetName                       | Specifies the net name that is to be assigned for the TMS pins.                                                                     | string      | false    |
| -<br>inputNetNamesOfInstanceChain | Specifies the list of net names to be assigned for the TDO-TDI chain pins.                                                          | string_list | false    |

## Examples

- `DefineJTAGChain -name Jtag -instanceList {U9 U1 U5} -inputNetNamesOfInstanceChain {U9_TDO U1_TDO} -tckNetName jtag_tck -tmsNetName jtag_tms`
- `DefineJTAGChain -name Jtag -instanceList {U9 U1 U5} -inputNetNamesOfInstanceChain {U9_TDO U1_TDO} -tckNetName jtag_tck -tmsNetName jtag_tms -loop close`

## Related Commands

[DeleteJTAGChain](#)

## DefinePROMChain

Creates a PROM chain in the current design.

### Return

bool

### Syntax

```
DefinePROMChain [-name prom_chain_name] -prom_list {list_of_prom_instance_names} -device_list {device_instances_list} -mode prom_mode [-
configuration type_of_configuration]
```

### Parameters

| Parameter          | Description                                                                                                                                                            | Type        | Optional |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -name              | Specifies the name that is to be specified for the PROM chain. FSP automatically assigns a name to the PROM chain in case no name is specified.                        | string      | true     |
| -prom_list         | Specifies a list of prom instance names that is to be connected in the PROM chain.                                                                                     | string_list | false    |
| -device_list       | Specifies a list of device instance names that is to be connected in the PROM chain.                                                                                   | string_list | false    |
| -mode              | Specifies the PROM chain mode. Valid values are serial or selectmap.                                                                                                   | string      | false    |
| -<br>configuration | Specifies the type of PROM chain configuration. Valid values are identical or different. By default, identical configuration is used in case no argument is specified. | string      | true     |

### Examples

- DefinePROMChain -name PROM1 -prom\_list [list U4 U5] -device\_list [list U1 U2] -mode serial -configuration identical
- DefinePROMChain -name PROM2 -prom\_list [list U9 U10] -device\_list [list U1 U2] -mode selectmap -configuration different

## Related Commands

[DeletePROMChain](#)

## DeleteAllDesignNetGroups

Deletes all the NetGroups in the design.

### Return

bool

### Syntax

```
DeleteAllDesignNetGroups
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
DeleteAllDesignNetGroups
```

## Related Commands

- [AutoNetGroupDesignGroupWise](#)
- [AutoNetGroupInterface](#)
- [AutoNetGroupGroupWise](#)
- [AutoNetGroupSwappablePins](#)

# DeleteAllInstanceCustomAttributes

Removes all the custom attributes from the specified instance.

## Return

bool

## Syntax

```
DeleteAllInstanceCustomAttributes instance_name
```

## Parameters

| Parameter     | Description                                                                                    | Type   | Optional |
|---------------|------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance from which the complete custom attributes is to be deleted. | string | false    |

## Examples

```
DeleteAllInstanceCustomAttributes U2
```

## Related Commands

- [DeleteInstanceCustomAttribute](#)
- [GetInstanceCustomAttributeValue](#)
- [AddInstanceCustomAttribute](#)

# DeleteAllInstancePinCustomAttributes

Deletes the entire custom attribute that are attached to the specified instance's pin.

## Return

bool



## Syntax

```
DeleteAllInstancePinCustomAttributes instance_name pin_number
```

## Parameters

| Parameter     | Description                                                                                  | Type   | Optional |
|---------------|----------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                                                          | string | false    |
| pin_number    | Specifies the name of the instance's pins of which you want to delete the custom attributes. | string | false    |

## Examples

```
DeleteAllInstancePinCustomAttributes U2 G18
```

## Related Commands

- [DeleteInstancePinCustomAttribute](#)
- [GetInstancePinCustomAttributeValue](#)
- [AddInstancePinCustomAttribute](#)

## DeleteAllNetCustomAttributes

Deletes the complete custom attribute details from the specified net.

## Return

bool

## Syntax

```
DeleteAllNetCustomAttributes net_name
```

## Parameters

| Parameter | Description                                                                                      | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------|--------|----------|
| net_name  | Specifies the name of the net from which the complete custom attribute details is to be deleted. | string | false    |

## Examples

```
DeleteAllNetCustomAttributes XP1_DDR2_DQS9_N
```

## Related Commands

- [AddNetCustomAttribute](#)
- [DeleteNetCustomAttribute](#)

## DeleteInstance

Deletes an instance from the design.

## Return

bool

## Syntax

```
DeleteInstance {instance_name_list}
```

## Parameters

| Parameter          | Description                                       | Type              | Optional |
|--------------------|---------------------------------------------------|-------------------|----------|
| instance_name_list | List of the instance names that is to be deleted. | stringstring_list | false    |

## Examples

- `DeleteInstance U1`
- `DeleteInstance XP2`
- `DeleteInstance [list U1 U2 U3 U4]`

## Related Commands

- [AddPart](#)
- [AddPartOrCAD](#)
- [PlaceInstance](#)

# DeleteInstanceCustomAttribute

Removes the specified custom attribute from the specified instance.

## Return

bool

## Syntax

```
DeleteInstanceCustomAttribute instance_name key
```

## Parameters

| Parameter     | Description                                                                            | Type   | Optional |
|---------------|----------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance from which you want to delete the custom attribute. | string | false    |
| key           | Specifies the name of the key that is to be deleted from the specified instance.       | string | false    |

## Examples

```
DeleteInstanceCustomAttribute U2 part_description
```

## Related Commands

- [DeleteAllInstanceCustomAttributes](#)
- [AddInstanceCustomAttribute](#)
- [GetInstanceCustomAttributeValue](#)

## DeleteInstanceNets

Removes the connectivity/nets for the specified instance.

### Return

bool

### Syntax

```
DeleteInstanceNets instance_name
```

### Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| instance_name | Name of the instance for which the nets is to be removed. | string | no       |

### Examples

- `DeleteInstanceNets U1`
- `DeleteInstanceNets XP2`

## Related Commands

- [DeleteNets](#)
- [AddNet](#)
- [ClearNets](#)

## DeleteInstancePinCustomAttribute

Deletes the specified custom attribute from the specified instance's pin.

### Return

bool

### Syntax

```
DeleteInstancePinCustomAttribute
```

## Parameters

| Parameter     | Description                                                              | Type   | Optional |
|---------------|--------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                                      | string | false    |
| pin_number    | Specifies the pin number of which the custom attribute is to be deleted. | string | false    |
| key           | Specifies the name of the custom attribute that is to be deleted.        | string | false    |

## Examples

```
DeleteInstancePinCustomAttribute U2 G18 pin_description
```

## Related Commands

- [AddInstancePinCustomAttribute](#)
- [GetInstancePinCustomAttributeValue](#)

## DeleteInstanceSelectedPinNets

Removes all nets connected to selected pins of given instance

## Return

bool

## Syntax

```
DeleteInstanceSelectedPinNets instanceName
```

## Parameters

| Parameter    | Description           | Type   | Optional |
|--------------|-----------------------|--------|----------|
| instanceName | Specify instance name | string | false    |

## Examples

- `DeleteInstanceSelectedPinNets U1`
- `DeleteInstanceSelectedPinNets XP1`

## Related Commands

No Related Commands

## DeleteJTAGChain

Deletes the specified JTAG chain from the design.

## Return

void

## Syntax

```
DeleteJTAGChain jtag_chain_name
```

## Parameters

| Parameter       | Description                                         | Type   | Optional |
|-----------------|-----------------------------------------------------|--------|----------|
| jtag_chain_name | Specifies the name of the JTAG chain to be deleted. | string | false    |

## Examples

```
DeleteJTAGChain Jtag
```

## Related Commands

- [DefineJTAGChain](#)
- [GetJTAGChains](#)

## DeleteNetCustomAttribute

Deletes specified custom attribute from the specified net.

## Return

bool

## Syntax

```
DeleteNetCustomAttribute net_name key
```

## Parameters

| Parameter | Description                                                                               | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------|--------|----------|
| net_name  | Specifies the name of the net from which the specified custom attribute is to be deleted. | string | false    |
| key       | Specifies the key of the custom attribute that is to be deleted.                          | string | false    |

## Examples

```
DeleteNetCustomAttribute net_type
```

## Related Commands

- [AddNetCustomAttribute](#)

## DeleteNetGroups

Deletes the specified NetGroups.

## Return

bool

## Syntax

```
DeleteNetGroups {net_group_list}
```

## Parameters

| Parameter      | Description                                          | Type        | Optional |
|----------------|------------------------------------------------------|-------------|----------|
| net_group_list | Specifies a list of NetGroups that is to be deleted. | string_list | false    |

## Examples

```
DeleteNetGroups {U1_U2_DATA_BYTE1 U1_U2_DATA_BYTE2}
```

## Related Commands

- [AutoNetGroupDesignGroupWise](#)
- [AutoNetGroupInterface](#)
- [AutoNetGroupGroupWise](#)
- [AutoNetGroupSwappablePins](#)

## DeleteNets

Deletes the specified nets from the design. Specify at least one command option. Command does not remove power regulator and fixed intern, extern net.

## Return

bool

## Syntax

```
DeleteNets [-all] [-clocks] [-constraint_pins] [-nets {list_of_net_names}] [-bus bus_name] [-inst_or_protocol instance_or_protocol_name] [-inst_or_protocol_group_or_bank group_or_bank_name]
```

## Parameters

| Parameter                       | Description                                                                                     | Type        | Optional |
|---------------------------------|-------------------------------------------------------------------------------------------------|-------------|----------|
| -all                            | Deletes all the nets.                                                                           | bool        | true     |
| -clocks                         | Deletes all the clock nets.                                                                     | bool        | true     |
| -constraint_pins                | Deletes all the constraint pin nets.                                                            | bool        | true     |
| -nets                           | Deletes all the specified nets.                                                                 | string_list | true     |
| -bus                            | Deletes all the nets belonging to specified bus name.                                           | string      | true     |
| -inst_or_protocol               | Deletes all the nets belonging to specified interface or protocol.                              | string      | true     |
| -inst_or_protocol_group_or_bank | Deletes all the nets belonging to specified group (or bank) of specified interface or protocol. | string      | true     |

## Examples

- `DeleteNets -all`
- `DeleteNets -clocks`
- `DeleteNets -clocks -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `DeleteNets -clocks -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `DeleteNets -clocks -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `DeleteNets -constraint_pins`
- `DeleteNets -constraint_pins -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `DeleteNets -constraint_pins -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `DeleteNets -constraint_pins -inst_or_protocol U1 U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `DeleteNets -nets [list XP1_DDR_A0 XP1_DDR_A1]`
- `DeleteNets -bus XP1_DDR_A`
- `DeleteNets -inst_or_protocol U1`
- `DeleteNets -inst_or_protocol U1_U3`
- `DeleteNets -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `DeleteNets -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `DeleteNets -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`

## Related Commands

- [ClearNets](#)

## DeletePowerRegulator

Removes the specified power regulator from the design.

### Return

bool

### Syntax

```
DeletePowerRegulator regulator_name
```

### Parameters

| Parameter      | Description                                        | Type   | Optional |
|----------------|----------------------------------------------------|--------|----------|
| regulator_name | Name of the power regulator that is to be deleted. | string | false    |

## Examples

```
DeletePowerRegulator V_0_9
```

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## DeletePowerRegulators

Removes the specified power regulators from the design.

### Return

bool

### Syntax

```
DeletePowerRegulator {regulator_name_list}
```

### Parameters

| Parameter           | Description                                                              | Type        | Optional |
|---------------------|--------------------------------------------------------------------------|-------------|----------|
| regulator_name_list | Specifies a list of names of the power regulators that is to be deleted. | string_list | false    |

### Examples

- `DeletePowerRegulators [GetPowerRegulators]`
- `DeletePowerRegulators {V_0_9 V_1_8}`

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)



## DeletePROMChain

Deletes the specified PROM chain from the design.

### Return

bool

### Syntax

```
DeletePROMChain prom_chain_name
```

### Parameters

| Parameter       | Description                                                 | Type   | Optional |
|-----------------|-------------------------------------------------------------|--------|----------|
| prom_chain_name | Specifies the name of the PROM chain that is to be deleted. | string | false    |

### Examples

```
DeletePROMChain PROM_Chain1
```

### Related Commands

[DefinePROMChain](#)

## DeleteTermination

Removes the specified termination from the design.

### Return

bool

### Syntax

```
DeleteTermination termination_name
```

### Parameters

| Parameter        | Description                                                  | Type   | Optional |
|------------------|--------------------------------------------------------------|--------|----------|
| termination_name | Specifies the name of the termination that is to be removed. | string | false    |

### Examples

- `DeleteTermination ser_term`
- `DeleteTermination power_fldr`

## Related Commands

- [AddTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [SpecifyDiffPinTermination](#)

## DifferSpecialPurposePinsUsage

Specifies whether to reserve special purpose pins such as VREF, VRP, VRN, CLOCK while allocating pins in synthesis to the possible extent. By default, this option is set to true, which means during synthesis special purpose pins will be deferred for allocation to the possible extent.

### Return

void

### Syntax

```
DifferSpecialPurposePinsUsage device_instance_name value_to_differ_or_not
```

### Parameters

| Parameter              | Description                                      | Type   | Optional |
|------------------------|--------------------------------------------------|--------|----------|
| device_instance_name   | Specifies the name of the device instance.       | string | false    |
| value_to_differ_or_not | Specifies the value. The default value is false. | bool   | false    |

### Examples

- `DifferSpecialPurposePinsUsage U1 true`
- `DifferSpecialPurposePinsUsage U2 false`

## Related Commands

[GetDeviceInstanceList](#)

## DoubleListTest

Tests the double list variable type.

### Return

double\_list

### Syntax

```
DoubleListTest arg
```

## Parameters

| Parameter | Description                                                    | Type        | Optional |
|-----------|----------------------------------------------------------------|-------------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | double_list | false    |

## Examples

- `DoubleListTest [list 90.89 101.2]`
- `DoubleListTest {90.89 636.75 537.846}`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

## DoubleTest

Tests the double variable type.

### Return

double

### Syntax

```
DoubleTest arg
```

## Parameters

| Parameter | Description                                                          | Type   | Optional |
|-----------|----------------------------------------------------------------------|--------|----------|
| arg       | Specifies the value of the variable type value that is to be tested. | double | false    |

## Examples

- `DoubleTest 89.0`
- `DoubleTest 87.9363`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

## env

Returns a list of environment variables. This list includes both FSP and system variables.

## Return

string\_list

## Syntax

```
env
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
env
```

## Related Commands

- [GetEnvVariable](#)
- [SetEnvVariable](#)

## ExportConstraints

Exports the constraints in an external file for the specified device instance.

## Return

bool

## Syntax

```
ExportConstraints -d device_instance_name -f file_path -p partial_or_full_export
```

## Parameters

| Parameter | Description                                                                                                                                                                                            | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device instance for which the constraints is to be exported.                                                                                                                 | string | false    |
| -f        | Specifies the complete file path in which the constraints is to be saved.                                                                                                                              | string | false    |
| -p        | Specifies whether partial constraints or full constraints to be exported. Valid values are true and false. True indicates to export partial constraint export and false to export all the constraints. | bool   | false    |

## Examples

- `ExportConstraints -d U1 ./output/U1.ucf -p false`
- `ExportConstraints -d U2 ./output/U2.tcl -p false`

## Related Commands

- [GeneratePlanAheadScripts](#)
- [GenerateConstraintFiles](#)
- [ImportConstraints](#)

# ExportCSVFromConnectorMapping

Exports the connector mapping data to the specified file path.

## Return

bool

## Syntax

```
ExportCSVFromConnectorMapping -a export_file_path -i instance_name
```

## Parameters

| Parameter | Description                                                        | Type   | Optional |
|-----------|--------------------------------------------------------------------|--------|----------|
| -a        | Specifies the file path to export the connector mapping data.      | string | no       |
| -i        | Specifies the connector name to export the connector mapping data. | string | no       |

## Examples

```
ExportCSVFromConnectorMapping -a ./J1.csv -i J1
```

## Related Commands

[MapConnectorPinAssignment](#)

# ExportCSVfromDesignConnectivity

Exports the pin and connectivity information of the specified instance in an command separated value(csv) format.

## Return

bool

## Syntax

```
ExportCSVfromDesignConnectivity -f export_file_path [-d delimiter] [-i {instance_name_list}] [-c {export_column_list}] [-v
export_only_visible_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                           | Type        | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -f        | Specifies the csv file path to store the pin and connectivity information.                                                                                                                                            | string      | false    |
| -d        | Specifies the delimiter to be used for the CSV format. For example, comma),(pipe),;(semicolon),:(colon)                                                                                                               | string      | true     |
| -i        | Specifies the list of instance names of which the pin and connectivity information is to be exported. In case this argument is not specified, the command exports data for all the instances.                         | string_list | true     |
| -c        | Specifies a list of columns names list whose values is to be exported for the specified instances. In case this argument is not specified, the command exports data for all columns of design connectivity.           | string_list | true     |
| -v        | Specifies whether the data to be exported for the visible pins or for all the pins in design connectivity. In case this argument is not specified, the command exports data for all (visible and invisible) the pins. | bool        | true     |

## Examples

- `ExportCSVfromDesignConnectivity -f ./pin_view.csv`
- `ExportCSVfromDesignConnectivity -f ./pin_view.csv -d | -i {U1 U4}`
- `ExportCSVfromDesignConnectivity -f ./de_net_view.csv -d \t -i {U2} -c {"Instance/Protocol Name\" \"Pin/Port Name\" \"Pin Number\" \"Pin Type\"} -v`
- `ExportCSVfromDesignConnectivity -f ./pin_view.csv -d , -i {U2_U1 XP1_U1_U2} -c {"Pin Number\" \"Pin/Port Name\" \"Pin Type\"}`

## Related Commands

- [SetDesignConnectivityView](#)
- [ImportCSVInDesignConnectivity](#)

## ExportCSVfromDesignExplorer

Exports the pin and connectivity information of the specified instance in an command separated value(csv) format.

## Return

bool

## Syntax

```
ExportCSVfromDesignExplorer -f export_file_path -d delimiter [-i {instance_name_list}] [-c {export_column_list}] [-v
export_only_visible_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                           | Type        | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -f        | Specifies the csv file path to store the pin and connectivity information.                                                                                                                                            | string      | no       |
| -d        | Specifies the delimiter to be used for the CSV format. For example, comma),(pipe),(semicolon),(colon)                                                                                                                 | string      | no       |
| -i        | Specifies the list of instance names of which the pin and connectivity information is to be exported. In case this argument is not specified, the command exports data for all the instances.                         | string_list | yes      |
| -c        | Specifies a list of columns names list whose values is to be exported for the specified instances. In case this argument is not specified, the command exports data for all columns of design connectivity.           | string_list | yes      |
| -v        | Specifies whether the data to be exported for the visible pins or for all the pins in design connectivity. In case this argument is not specified, the command exports data for all (visible and invisible) the pins. | bool        | yes      |

## Examples

- `ExportCSVfromDesignExplorer -f ./pin_view.csv -d ,`
- `ExportCSVfromDesignExplorer -f ./pin_view.csv -d t -i {U1 U4}`
- `ExportCSVfromDesignExplorer -f ./de_net_view.csv -d t -i {U2} -c {"Instance/Protocol Name" "Pin/Port Name" "Pin Number" "Pin Type"} -v`
- `ExportCSVfromDesignExplorer -f ./pin_view.csv -d , -i {U2_U1 XP1_U1_U2} -c {"Pin Number" "Pin/Port Name" "Pin Type"}`

## Related Commands

- [SetDesignExplorerView](#)
- [ImportCSVInDesignExplorer](#)

## ExportCSVFromFPGAPort

Exports the values of the RTL ports names and Assign to pin columns for all the pnis that are targeted to the specified FPGA.

## Return

bool

## Syntax

```
ExportCSVFromFPGAPort -i device_name -a csv_file_path
```

## Parameters

| Parameter | Description                                                                         | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------|--------|----------|
| -a        | Specifies the csv file path that is to be used to save the RTL ports data.          | string | no       |
| -i        | Specifies the name of the device instance of which the RTL ports is to be exported. | string | no       |

## Examples

```
ExportCSVFromFPGAPort -i U1 -a ./ul_fpga_port_data.csv
```

## Related Commands

- [UpdateFPGAPortsFromCSV](#)
- [MapPortNamestoPinNames](#)

## ExportCSVFromInstancePart

Exports the pin definition of the specified interface from rules file to the CSV file.

### Return

bool

### Syntax

```
ExportCSVFromInstancePart -p interface_name -a csv_file_path [-c {export_column_name_list}]
```

### Parameters

| Parameter | Description                                                                                                                                      | Type        | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -a        | Specifies the CSV file path that is to be used to save the definition of the specified instance.                                                 | string      | false    |
| -p        | Specifies the name of the instance of which the definitions is to be exported.                                                                   | string      | false    |
| -c        | Specifies the list of the column names that is to be exported. In case argument is not specified, then command will export data for all columns. | string_list | true     |

### Examples

- `ExportCSVFromInstancePart -p U1 -a ./export_U1.csv -c {"Pin Number\" \"Pin Name\" \"Diff. Type\"}`
- `ExportCSVFromInstancePart -p U1 -a ./export_U1.csv`

## Related Commands

[UpdateInstancePartFromCSV](#)

## ExportCSVFromPart

Exports the definition of the specified part to the CSV file.

### Return

bool

### Syntax

```
ExportCSVFromPart -p part_name -c csv_file_path [-g is_connector]
```



## Parameters

| Parameter | Description                                                                                                                                                                                                                                       | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -p        | Specifies the name of the part of which the definitions is to be exported.                                                                                                                                                                        | string | false    |
| -c        | Specifies the csv file path where the definition of the part is to be saved.                                                                                                                                                                      | string | false    |
| -g        | Specifies whether the specified rules file of type connector or fixed pin interface. Use 'y' for generate connector rules file and 'n' for interface rules file. By default, this command considers specified rules file as interface rules file. | string | true     |

## Examples

- `ExportCSVFromPart -p ddr_dimm_x4_v4_v5 -c ./export_U1.csv`
- `ExportCSVFromPart -p ddr_dimm_x4_v4_v5 -c ./export_U1.csv -g y`

## Related Commands

- [CreatePartFromCSV](#)
- [UpdatePartFromCSV](#)

# ExportCSVFromProtocol

Outputs the definition of the specified protocol that is connected to the specified device in the CSV file.

## Return

bool

## Syntax

```
ExportCSVFromProtocol file_path protocol_name device_name
```

## Parameters

| Parameter     | Description                                                                       | Type   | Optional |
|---------------|-----------------------------------------------------------------------------------|--------|----------|
| file_path     | Specifies the csv file path where the CSV file is to be saved.                    | string | false    |
| protocol_name | Specifies the name of the protocol.                                               | string | false    |
| device_name   | Specifies the name of the device for which protocol definition is to be exported. | string | false    |

## Examples

- `ExportCSVFromProtocol ./protocol_export.csv U2_U1 U1`

## Related Commands

- [CreateProtocolFromCSV](#)
- [UpdateProtocolFromCSV](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromLibraryModel](#)

## ExportCSVFromVI

Outputs the definition of the virtual interface in the CSV file.

### Return

bool

### Syntax

```
ExportCSVFromVI file_path vi_name
```

### Parameters

| Parameter | Description                                                                               | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------|--------|----------|
| file_path | Specifies the file path where the CSV file need to be saved.                              | string | false    |
| vi_name   | specifies the name of the virtual interface of which the pin definition need to exported. | string | false    |

### Examples

```
ExportCSVFromVI ./export_vi.csv U2_VI2
```

### Related Commands

- [CreateVIFromCSV](#)
- [UpdateVIFromCSV](#)

## ExportCSVSchematicSymbolEditor

Exports the pin definition of the specified symbol from Schematic Symbol Editor to the CSV file.

### Return

bool

### Syntax

```
ExportCSVSchematicSymbolEditor -p interface_name -a csv_file_path [-c {export_column_name_list}]
```

### Parameters

| Parameter | Description                                                                                                                                          | Type        | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -a        | Specifies the CSV file path that is to be used to save the definition of the specified instance.                                                     | string      | false    |
| -p        | Specifies the name of the instance of which the definitions is to be exported.                                                                       | string      | false    |
| -c        | Specifies the list of the column names that is to be exported. In case argument is not specified, then the command will export data for all columns. | string_list | true     |

## Examples

- `ExportCSVSchematicSymbolEditor -p U1 -a ./export_U1.csv`
- `ExportCSVSchematicSymbolEditor -p U1 -a ./export_U1.csv -c {"Pin Number" "Location" "Pin Direction"}`

## Related Commands

[UpdateInstanceSymbolFromCSV](#)

# ExportDecaps

Exports Decaps info.

## Return

bool

## Syntax

```
ExportDecaps [-instances instance_name] -file file_name
```

## Parameters

| Parameter  | Description                                                                                                                                                                                  | Type        | Optional |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -instances | Decaps info for the specified instance name will be exported to specified file path. If instance name is not specified then, all decaps data associated with all instances will be exported. | string_list | true     |
| -file      | Specifies the export file path.                                                                                                                                                              | string      | false    |

## Examples

- `ExportDecaps -instances [list U1 U2] -file ./decaps.xml`
- `ExportDecaps -instances U1 -file ./decaps.xml`
- `ExportDecaps -file ./decaps.xml`

## Related Commands

[ImportDecaps](#)

# ExportDesignBlockSymbolCSV

Exports the pin definition of the specified symbol from Schematic Block Symbol Editor to the CSV file.

## Return

bool

## Syntax

```
ExportDesignBlockSymbolCSV -type [full|split] -a csv_file_path [-c {export_column_name_list}]
```

## Parameters

| Parameter | Description                                                                                                                                          | Type        | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -type     | Specifies the type of symbol such as full or split to be generated.                                                                                  | string      | false    |
| -a        | Specifies the CSV file path that is to be used to save the definition symbol.                                                                        | string      | false    |
| -c        | Specifies the list of the column names that is to be exported. In case argument is not specified, then the command will export data for all columns. | string_list | true     |

## Examples

- `ExportDesignBlockSymbolCSV -type split -a ./export_U1.csv`
- `ExportDesignBlockSymbolCSV -type full -a ./export_U1.csv -c {"Port Name" "Location" "Pin Direction"}`

## Related Commands

[GenerateDesignBlockSymbol](#)

# ExportDesignForPinSwap

Creates a backup of the current design in the specified path. The backup of the design can be used with Allegro for pin swap, placement, and reference designator sync.

## Return

bool

## Syntax

```
ExportDesignForPinSwap complete_new_fsp_database_file_path
```

## Parameters

| Parameter            | Description                                                                                          | Type   | Optional |
|----------------------|------------------------------------------------------------------------------------------------------|--------|----------|
| copyDatabaseFilePath | Specifies the name and path (absolute or relative) where the current design database is to be saved. | string | false    |

## Examples

- `ExportDesignForPinSwap ./design_copy.fsp`
- `ExportDesignForPinSwap /export/home/user/projects/my_project/design_copy.fsp`

## Related Commands

[ExportDesignForPinSwap](#)

# ExportDeviceConstraints

Exports the constraints in an external file for the specified device instance.

## Return

bool

## Syntax

```
ExportDeviceConstraints -d device_instance_name -f file_path -p partial_or_full_export
```

## Parameters

| Parameter | Description                                                                                                                                                                                            | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device instance for which the constraints is to be exported.                                                                                                                 | string | no       |
| -f        | Specifies the complete file path in which the constraints is to be saved.                                                                                                                              | string | no       |
| -p        | Specifies whether partial constraints or full constraints to be exported. Valid values are true and false. True indicates to export partial constraint export and false to export all the constraints. | bool   | no       |

## Examples

- `ExportDeviceConstraints -d U1 ./output/U1.ucf -p false`
- `ExportDeviceConstraints -d U2 ./output/U2.tcl -p false`

## Related Commands

- [GeneratePlanAheadScripts](#)
- [GenerateConstraintFiles](#)
- [ImportConstraints](#)

## ExportPDF

Exports the FSP design canvas view in a pdf format.

## Return

void

## Syntax

```
ExportPDF absoluteFilePath
```

## Parameters

| Parameter        | Description                                  | Type   | Optional |
|------------------|----------------------------------------------|--------|----------|
| absoluteFilePath | The path and name of the pdf file to export. | string | false    |

## Examples

```
ExportPDF d:/fsp_design.pdf
```

## Related Commands

[CloseProject](#)

## ExportPinAssignemntsForConnector

Exports the pin number and net name details in a text file for the specified connector.

### Return

bool

### Syntax

```
ExportPinAssignemntsForConnector instance_name filePath
```

### Parameters

| Parameter     | Description                                                                                          | Type   | Optional |
|---------------|------------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the connector for which the pin number and net name details is to be exported. | string | false    |
| filePath      | Specifies the name of the file path where the text file need to saved.                               | string | false    |

### Examples

```
ExportPinAssignemntsForConnector J1 [GetProjectFilePath]/j1_netlist.txt
```

## Related Commands

- [ImportPinAssignmentsForConnector](#)
- [GenerateNetList](#)

## ExportPinAssignmentsForConnector

Exports the pin number and net name details in a text file for the specified connector.

### Return

bool

### Syntax

```
ExportPinAssignmentsForConnector instance_name filePath
```

### Parameters

| Parameter     | Description                                                                                          | Type   | Optional |
|---------------|------------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the connector for which the pin number and net name details is to be exported. | string | false    |
| filePath      | Specifies the name of the file path where the text file need to saved.                               | string | false    |

## Examples

```
ExportPinAssignmentsForConnector J1 [GetProjectFilePath]/j1_netlist.txt
```

## Related Commands

- [ImportPinAssignmentsForConnector](#)
- [GenerateNetList](#)

# ExportPlacementData

Exports the component placement data to the XML file.

## Return

bool

## Syntax

```
ExportPlacementData -e export_file_path
```

## Parameters

| Parameter | Description                                                                          | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------|--------|----------|
| -e        | Specifies the XML file path that is to be used to save the component placement data. | string | false    |

## Examples

```
ExportPlacementData -e ./placement.xml
```

## Related Commands

- [ImportPlacementXMLFile](#)
- [GenerateDEHDLschematics](#)
- [GenerateLayoutData](#)
- [UpdateLayoutData](#)
- [GenerateOrCADschematics](#)

# ExportProtocolDefinition

Exports the definition of the specified protocol to an external file.

## Return

bool

## Syntax

```
ExportProtocolDefinition protocol_name protocolDefinitionFilePath
```

## Parameters

| Parameter                  | Description                                                                                       | Type   | Optional |
|----------------------------|---------------------------------------------------------------------------------------------------|--------|----------|
| protocol_name              | Specifies a protocol name (devices in comma separated format) whose definition is to be exported. | string | false    |
| protocolDefinitionFilePath | Specifies the absolute or relative path where the protocol definition to be exported.             | string | false    |

## Examples

- `ExportProtocolDefinition U3,U4 [GetProjectFilePath]/U3_U4_protocol.prf`
- `ExportProtocolDefinition U8,U9,U10,U11 [GetProjectFilePath]/data_bus_protocol.prf`

## Related Commands

- [GetProtocolNames](#)
- [RenameProtocol](#)
- [CreateProtocolFromExistingProtocol](#)

## ExportSpreadSheetFiles

Exports the database of the design for each devices in a Comma Separated Values(CSV) file at the \$project\_dir/output/spreadsheet directory.

### Return

bool

### Syntax

```
ExportSpreadSheetFiles
```

### Parameters

No Parameters

### Examples

```
ExportSpreadSheetFiles
```

### Related Commands

No Related Commands

## ExportVirtualInterfaceDefinition

Exports the definition of the virtual interface to an external file.

### Return

bool



## Syntax

```
ExportVirtualInterfaceDefinition virtualInterfaceName viDefinitionFilePath
```

## Parameters

| Parameter            | Description                                                                                     | Type   | Optional |
|----------------------|-------------------------------------------------------------------------------------------------|--------|----------|
| virtualInterfaceName | Specifies the name of the virtual interface instance whose pin definition is to be exported.    | string | false    |
| viDefinitionFilePath | Specify absolute or relative virtual interface file path where definition needs to be exported. | string | false    |

## Examples

- `ExportVirtualInterfaceDefinition U4_VI22 [GetProjectFilePath]/U4_VI22_definition.vrf`
- `ExportVirtualInterfaceDefinition DSA_KEYBOARD_VI /export/home/fsp/projects/rules/keyboard_vi.vrf`

## Related Commands

[CreateVIFromExistingVI](#)

# FlipInstance

Flips the specified instance to the specified side.

## Return

bool

## Syntax

```
FlipInstance {instance_name_list} side
```

## Parameters

| Parameter     | Description                                                                                        | Type        | Optional |
|---------------|----------------------------------------------------------------------------------------------------|-------------|----------|
| instance_name | List of the instance names that is to be flipped.                                                  | string_list | false    |
| side          | Specifies the side to which the instance needs to be flipped. The valid values are top and bottom. | string      | false    |

## Examples

- `FlipInstance U5 bottom`
- `FlipInstance U5 top`
- `FlipInstance [list U1 U2 U3] top`

## Related Commands

- [GetInstanceSide](#)
- [RotateInstance](#)

## GenerateAllegroSymbol

Generates the DE HDL symbol for the specified instance.

### Return

bool

### Syntax

```
GenerateAllegroSymbol -i instance_name [-s symbolName] [-l location] [-p]
```

### Parameters

| Parameter | Description                                                                                                                                        | Type   | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the instance for which the symbol is to be generated.                                                                        | string | false    |
| -s        | Specifies the name of the symbol to be generated. In case argument is not specified, the command uses the existing symbol name.                    | string | true     |
| -l        | Specifies the name of library where the symbols is to be generated. In case argument is not specified, the command uses the existing library name. | string | true     |
| -p        | Specifies whether the global power pin property is to be generated for the symbol. Default value is false.                                         | bool   | true     |

### Examples

- `GenerateAllegroSymbol -i XP1 -l fsp_fe_lib`
- `GenerateAllegroSymbol -i XP1 -l fsp_fe_lib -p`

### Related Commands

[GenerateDEHDL Schematics](#)

## GenerateASADesign

Generates ASA Design files.

### Return

bool

### Syntax

```
GenerateASADesign [-generate_fpga_hier_blocks] [-generate_net_groups]
```

### Parameters

| Parameter                  | Description                                                                                                                                                      | Type | Optional |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|----------|
| -generate_fpga_hier_blocks | Specifies whether to generate hierarchical schematics for FPGAs under root block. FSP will generate flat schematics under root in case no argument is specified. | bool | true     |
| -generate_net_groups       | Specifies whether to propagate FSP defined net groups into Constraint Manager. Net group information will not be propagated in case no argument is specified.    | bool | true     |

## Examples

- `GenerateASADesign -generate_fpga_hier_blocks -generate_net_groups`
- `GenerateASADesign`

## Related Commands

- [GenerateAllegroSymbol](#)
- [GenerateDEHDL Schematics](#)

# GenerateConstraintFiles

Generates the constraints files for all FPGAs in the design.

## Return

bool

## Syntax

```
GenerateConstraintFiles
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GenerateConstraintFiles
```

## Related Commands

- [GeneratePlanAheadScripts](#)
- [ExportConstraints](#)
- [ImportConstraints](#)

# GenerateDEHDL Schematics

Generates the DE HDL schematics.

## Return

bool

## Syntax

```
GenerateDEHDL Schematics [-p] [-t] [-actual_port_type] [-flat_hier_terms] [-flat_schematics] [-skip_unused] [-donot_mix_hier_symbols] [-donot_mix_instance_symbols] [-net_name_as_pin_name] [-net_group]
```

## Parameters

| Parameter                             | Description                                                                                                                                                                               | Type | Optional |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|----------|
| -preserve                             | Specifies whether to generate schematics in preserve mode. The command generates schematics in non-preserve mode in case no argument is specified.                                        | bool | true     |
| -generate_fpga_hier_blocks            | Specifies whether to generate hierarchical schematics for FPGAs under root block. FSP will generate flat schematics under root in case no argument is specified.                          | bool | true     |
| -use_actual_port_type                 | Specifies whether to use actual port direction for hierarchal symbol port. FSP uses 'Inout' as port type in case no argument is specified.                                                | bool | true     |
| -flatten_termination_hier_blocks      | Specifies whether to place underlying discrete components directly in schematics for hierarchal terminations block. FSP will place hierarchal block in case no argument is specified.     | bool | true     |
| -skip_unused_splits                   | Specifies whether to exclude symbol splits that have no connections. FSP will place all symbol splits in schematics in case no argument is specified.                                     | bool | true     |
| -donot_mix_hier_symbols               | Specifies whether to mix hierarchal block symbols with primitive symbols. By default, hierarchal symbols are placed with the primitive symbols in case no argument is specified.          | bool | true     |
| -donot_mix_different_instances_symbol | Specifies whether to use a unique schematic pages for each instance. In case argument is not specified, symbols of different instances will be placed together.                           | bool | true     |
| -show_net_name_as_instance_pin_name   | Specifies whether to use the net name connected to the pin instead of FPGA symbol pin name. In schematics the symbol pin name will be shown as pin name in case no argument is specified. | bool | true     |
| -generate_net_groups                  | Specifies whether to propagate FSP defined net groups into Constraint Manager. Net group information will not be propagated in case no argument is specified.                             | bool | true     |

## Examples

- `GenerateDEHDLschematics -p -t -actual_port_type -flat_hier_terms -flat_schematics -skip_unused -donot_mix_hier_symbols -donot_mix_instance_symbols -net_name_as_pin_name -net_group`
- `GenerateDEHDLschematics -p`
- `GenerateDEHDLschematics -t -donot_mix_hier_symbols`
- `GenerateDEHDLschematics -net_group`
- `GenerateDEHDLschematics -p -flat_hier_terms -net_group`
- `GenerateDEHDLschematics`

## Related Commands

[GenerateAllegroSymbol](#)

# GenerateDesignBlockSymbol

Updates DE-HDL design block symbol (full or splits) by importing the CSV file (if specified).

## Return

bool

## Syntax

```
GenerateDesignBlockSymbol -type [full|split] [-p] [[-c csv_file_path] -m column_mapping -r reference_column [-d delimiter] [-i ignore_rows]]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -type     | Specifies the type of symbol such as full or split to be generated.                                                                                            | string            | false    |
| -p        | Specifies if symbol needs to be generated in preseve graphics mode. By default, this command generates symbol in non-preserve graphics mode.                   | bool              | true     |
| -c        | Specifies the path of the csv file that is to be used to update the instance symbol pin properties.                                                            | string            | true     |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | true     |
| -r        | Specifies the name of the reference column.                                                                                                                    | string            | true     |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), '(space) etc...                                | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

- `GenerateDesignBlockSymbol -type full`
- `GenerateDesignBlockSymbol -type split`
- `GenerateDesignBlockSymbol -type split -p`
- `GenerateDesignBlockSymbol -type full -c ./update_instance_symbol_csv.csv -d , -i 1 -r \"Port Name\" -m {\"Port Name\" 1 \"Location\" 2 \"Pin Direction\" 3}`
- `GenerateDesignBlockSymbol -type split -p -c ./update_instance_symbol_csv.csv -d , -i 1 -r \"Port Name\" -m {\"Port Name\" 1 \"Location\" 2 \"Pin Direction\" 3}`

## Related Commands

[ExportDesignBlockSymbolCSV](#)

# GenerateLayoutData

Generates placement and layout data required for Allegro design.

## Return

bool

## Syntax

```
GenerateLayoutData {instance_list}
```

## Parameters

| Parameter     | Description                                                                                                                                                                      | Type        | Optional |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| instance_list | Specifies the list of instance names of which you want to generate the layout data. For this command to run successfully, the specified instances must be present on the canvas. | string_list | false    |

## Examples

```
GenerateLayoutData {U1 XP1}
```

## Related Commands

- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)
- [UpdateLayoutData](#)

## GenerateNCReport

Enables or disables the generation of report for unconnected signals during synthesis. The Synthesis Failure Report provides you a detail explanation and reason for the failed connections.

## Return

bool

## Syntax

```
GenerateNCReport isGenerate
```

## Parameters

| Parameter  | Description                                                                                                            | Type | Optional |
|------------|------------------------------------------------------------------------------------------------------------------------|------|----------|
| isGenerate | Specifies the value to enable or disable the Synthesis Failure Report generation. The valid values are true and false. | bool | no       |

## Examples

- ```
GenerateNCReport true
```
- ```
GenerateNCReport false
```

## Related Commands

[GenerateNCReportStatus](#)

## GenerateNCReportStatus

Returns the status of Synthesis Failure Report flag. Returns 1, if flag is on and 0 for off.

## Return

bool

## Syntax

```
GenerateNCReportStatus
```

## Parameters

No Parameters

## Examples

GenerateNCReportStatus

## Related Commands

[GenerateNCReport](#)

# GenerateNetgroupRatbundle

Generates rat bundles between the specified instances

## Return

bool

## Syntax

```
GenerateNetgroupRatbundle [-p instancePairMap] -b allegro_board_file_name -r output_script_file_name
```

## Parameters

| Parameter | Description                                                                   | Type              | Optional |
|-----------|-------------------------------------------------------------------------------|-------------------|----------|
| -p        | Specifies the list of instance pairs.                                         | string_string_map | true     |
| -b        | Specifies the board file path that is to be used to extract the rats bundles. | string            | false    |
| -r        | Specifies the Allegro script name to generate commands to update rats bundle. | string            | false    |

## Examples

- GenerateNetgroupRatbundle -p {LA3 \\"MUX1:2\\" LA\* \\"MUX11\\"} -b D:/testDesign/project1.brd -r LA1.scr
- GenerateNetgroupRatbundle -p {LA3 \\"MUX8 MUX9 MUX10\\" LA4 \\"MUX11 MUX12 MUX13\\"} -b ../physical/project1.brd -r DUT.scr

## Related Commands

[CreateTargetDeviceSet](#)

# GenerateNetList

Generates the netlist for the current design to the \$project\_dir/output/netlist.txt.

## Return

bool

## Syntax

```
GenerateNetList
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

GenerateNetList

## Related Commands

- [ChangeNetName](#)
- [ClearNets](#)

# GenerateOrCADSchematics

Generates the OrCAD schematics.

## Return

bool

## Syntax

```
GenerateOrCADSchematics [-f fsp_schematics_name] [-l is_create_root_schematics] [-n {instance_name_ist}] [-o schematic_output_path] [-p project_name] [-r root_design_name] [-s is_skip_unused_splits] [-t termination_on_seperate_page]] [-b bit_order]
```

## Parameters

| Parameter | Description                                                                                                                                                                              | Type        | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -f        | Specifies the name of the schematic file (.dsn) that is to be generated. In case argument is not specified, the command uses existing .dsn file set for project.                         | string      | true     |
| -r        | Specifies the name of the root design. In case argument is not specified, the command uses existing root design set for project.                                                         | string      | true     |
| -p        | Specifies the name of the schematic project.                                                                                                                                             | string      | true     |
| -o        | Specifies the directory path where the schematic is to be generated.                                                                                                                     | string      | true     |
| -n        | Specifies the list of instance names for which the schematics is to be generated. In case argument is not specified, the command generates schematics based on existing settings.        | string_list | true     |
| -l        | Specifies whether the root schematics is to be generated. In case argument is not specified, the command generates root schematics.                                                      | bool        | true     |
| -s        | Specifies whether the schematic pages is to be generated for unused splits of specified instances. In case argument is not specified, the command uses settings of the existing project. | bool        | true     |
| -t        | Specifies whether the terminations is to be generated on separate page. In case argument is not specified, the command generates termination on the same page of the connected instance. | bool        | true     |
| -b        | Specifies how the bus bit order should be displayed in generated description. The valid values are M2L (MSB to LSB) and L2M (LSB to MSB). Default value is M2L.                          | string      | true     |



## Examples

- `GenerateOrCADSchematics`
- `GenerateOrCADSchematics -f test_sch -r root -p fsp_proj -t 1 -l 1`
- `GenerateOrCADSchematics -f test_sch -r root -p fsp_proj -t 1 -l 1 -b M2L`

## Related Commands

[GenerateOrCADSymbol](#)

# GenerateOrCADSymbol

Generates the OrCAD symbol for the specifies instance.

## Return

bool

## Syntax

```
GenerateOrCADSymbol -i instance_name [-l symbol_library_path] [-p package_name]
```

## Parameters

| Parameter | Description                                                                                                                                                               | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the instance for which the symbol is to be generated.                                                                                               | string | false    |
| -l        | Specifies the name of the .olb in which the symbol is to be generated. In case argument is not specified, the command generates symbol in existing .olb set for instance. | string | true     |
| -p        | Specifies the name of the package of the symbol. In case argument is not specified, the command uses existing package name set for instance.                              | string | true     |

## Examples

- `GenerateOrCADSymbol -i U1 -p test`
- `GenerateOrCADSymbol -i U2`

## Related Commands

[GenerateOrCADSchematics](#)

# GeneratePinNumberMappingFile

Generates pin number mapping file for the specified FPGA.

## Return

bool

## Syntax

```
GeneratePinNumberMappingFile fpga_part_name dra_name
```

## Parameters

| Parameter      | Description                          | Type   | Optional |
|----------------|--------------------------------------|--------|----------|
| fpga_part_name | Specifies the name of the fpga part. | string | false    |
| dra_name       | Specifies dra name .                 | string | false    |

## Examples

```
GeneratePinNumberMappingFile ep4cgx150df27 ep4cgx150df27_dra
```

## Related Commands

[GetInstanceNameList](#)

# GeneratePlanAheadScripts

Generates PlanAhead scripts and supported data files and stores them in the \$project\_dir/output/planahead directory. Generated TCL script can be used to run the IO DRCs in PlanAhead for all Xilinx FPGAs.

## Return

bool

## Syntax

```
GeneratePlanAheadScripts
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GeneratePlanAheadScripts
```

## Related Commands

- [ExportConstraints](#)
- [GenerateConstraintFiles](#)
- [ImportConstraints](#)

# GenerateVerilogBrdDescFile

Generates the verilog board level file for the current design in the \$project\_dir/output/verilog directory.

## Return

bool

## Syntax

```
GenerateVerilogBrdDescFile
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GenerateVerilogBrdDescFile
```

## Related Commands

- [GeneratePlanAheadScripts](#)
- [ExportConstraints](#)
- [GenerateConstraintFiles](#)
- [ImportConstraints](#)

# GetAllContiguousSignal

Returns a list of contiguous signals of the specified interface's group.

## Return

string\_list

## Syntax

```
GetAllContiguousSignal instance_name groupName
```

## Parameters

| Parameter     | Description                          | Type   | Optional |
|---------------|--------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface. | string | false    |
| groupName     | Specifies the name of the group.     | string | false    |

## Examples

```
GetAllContiguousSignal XP2 Address_Control
```

## Related Commands

- [GetContiguousSignal](#)
- [SetContiguousSignal](#)

## GetAllDoNotConnect

Returns a list of pin numbers that are preserved in the specified instance.

### Return

string\_list

### Syntax

```
GetAllDoNotConnect instance_name
```

### Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

### Examples

- `GetAllDoNotConnect U3`
- `GetAllDoNotConnect XP2`

### Related Commands

- [GetDoNotConnect](#)
- [ResetAllDoNotConnect](#)
- [ResetDoNotConnectPin](#)
- [SetDoNotConnectPin](#)

## GetAllegroCPMFileName

Returns the path of the Allegro CPM file referenced by design.

### Return

string

### Syntax

```
GetAllegroCPMFileName
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetAllegroCPMFileName
```

## Related Commands

# GetAllLockedNetNames

Returns the names of the nets that are locked in the design.

## Return

string\_list

## Syntax

```
GetAllLockedNetNames
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetAllLockedNetNames
```

## Related Commands

- [LockNets](#)
- [UnlockNets](#)

# GetAllProcessOptionNames

Returns a list of all the process options, defined in the Process Option Editor.

## Return

string\_list

## Syntax

```
GetAllProcessOptionNames
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetAllProcessOptionNames
```

## Related Commands

- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [LoadProcessOptions](#)
- [SaveProcessOptions](#)
- [RemoveProcessOption](#)
- [RemoveAllProcessOptions](#)
- [RenameProcessOption](#)

## GetAllTargetDevice

Returns a list of device instances to which the specified interface's group is targeted.

### Return

string\_list

### Syntax

```
GetAllTargetDevice instance_name groupName
```

## Parameters

| Parameter     | Description                                                                         | Type   | Optional |
|---------------|-------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface.                                                | string | false    |
| groupName     | Specifies the name of the group. Group can be targeted or non-targeted to any FPGA. | string | false    |

## Examples

- `GetAllTargetDevice XP2 Address_Control`
- `GetAllTargetDevice U3 data_nibble2`

## Related Commands

- [GetTargetDevice](#)
- [GetTargetInstanceListForDevice](#)
- [TargetDevice](#)
- [TargetToMultipleDevices](#)

## GetAllUseBanks

Returns list of all bank names that can be set for use bank for given group of interface instance. Returns empty list in case group is not targeted to any device.

### Return

string\_list

## Syntax

```
GetAllUseBanks instance_name groupName
```

## Parameters

| Parameter     | Description                          | Type   | Optional |
|---------------|--------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface. | string | false    |
| groupName     | Specifies the name of the group.     | string | false    |

## Examples

- `GetAllUseBanks XP2 Address_Control`
- `GetAllUseBanks U3 data_nibble2`

## Related Commands

- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

## GetApplyNetNameTemplate

Returns True if the apply net name template option is ON else returns False.

## Return

bool

## Syntax

```
GetApplyNetNameTemplate
```

## Parameters

No Parameters

## Examples

```
GetApplyNetNameTemplate
```

## Related Commands

- [SetDesignNetNameTemplate](#)
- [SetDesignProtocolNetNameTemplate](#)

## GetAvailableIOPinCount

Returns a number of IO pins count that are available for the connection in the specified device instance.

### Return

int

### Syntax

```
GetAvailableIOPinCount deviceInstanceName
```

### Parameters

| Parameter          | Description                                                                 | Type   | Optional |
|--------------------|-----------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device whose available IO pins count is required. | string | no       |

### Examples

- `tclGetAvailableIOPinCount U3`
- `tclGetAvailableIOPinCount U1`

## Related Commands

- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinCount](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinCount](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

## GetAvailableIOPinNumberList

Returns a list of IO pin numbers that are available for connections for the specified device instance.

### Return

string\_list

### Syntax

```
GetAvailableIOPinNumberList device_instance_name
```



## Parameters

| Parameter            | Description                                                                               | Type   | Optional |
|----------------------|-------------------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance of which available IO pin numbers are required. | string | false    |

## Examples

- `GetAvailableIOPinNumberList U3`
- `GetAvailableIOPinNumberList U1`

## Related Commands

- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

## GetAvailableLicenses

Returns a list of available FSP licenses.

### Return

string

### Syntax

```
GetAvailableLicenses
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetAvailableLicenses
```

## Related Commands

- [GetAvaliableLicenseType](#)
- [GetCurrentLicenseType](#)
- [SetLicenseType](#)

## GetAvailableNetGroupNames

Returns a list of NetGroups defined in the design. Use the -i option to restrict the scope of the command to the specified interface, protocol, or virtual interface.

### Return

string\_list

## Syntax

```
GetAvailableNetGroupNames [-i interface_or_protocol_or_vi_name]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                                            | Type   | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface, protocol, or virtual interface for which the NetGroups are required. If this argument is not specified, the NetGroups for the entire interface, protocol, or virtual interface will be returned . | string | true     |

## Examples

- `GetAvailableNetGroupNames`
- `GetAvailableNetGroupNames -i XP1`
- `GetAvailableNetGroupNames -i U1_U2`

## Related Commands

[GetConnectedNetGroupNames](#)

# GetAvaliableLicenseType

Returns a list of available FSP licenses. Note that it does not consider non-availability of license due to license consumed by another user.

## Return

string\_list

## Syntax

```
GetAvaliableLicenseType
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetAvaliableLicenseType
```

## Related Commands

- [GetAvailableLicenses](#)
- [GetCurrentLicenseType](#)
- [SetLicenseType](#)

# GetBankAvailableIOPinCount

Returns a number of IOs available for connection in the specified bank of the device instance.

## Return

int

## Syntax

```
GetBankAvailableIOPinCount deviceInstanceName bankName
```

## Parameters

| Parameter          | Description                                                      | Type   | Optional |
|--------------------|------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device.                                | string | no       |
| bankName           | Specifies the name of the bank whose available IOs are required. | string | no       |

## Examples

- `GetBankAvailableIOPinCount U3 1`
- `GetBankAvailableIOPinCount U10 Bank1`

## Related Commands

- [GetAvailableIOPinCount](#)
- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinCount](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

# GetBankAvailableIOPinNumberList

Returns a list of IO pin numbers that are available for connection in the specified bank of the device instance.

## Return

string\_list

## Syntax

```
GetBankAvailableIOPinNumberList deviceInstanceName bankName
```

## Parameters

| Parameter          | Description                                                                             | Type   | Optional |
|--------------------|-----------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance.                                              | string | false    |
| bankName           | Specifies the name of the bank for which the available IO pin numbers list is required. | string | false    |

## Examples

- `GetBankAvailableIOPinNumberList U3 1`
- `GetBankAvailableIOPinNumberList U10 Bank1`

## Related Commands

- [GetAvailableIOPinNumberList](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

## GetBankIOPinCount

Returns the number of IO pin count available in the specified bank of the device instance.

### Return

int

### Syntax

```
GetBankIOPinCount deviceInstanceName bankName
```

### Parameters

| Parameter          | Description                                                    | Type   | Optional |
|--------------------|----------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance.                     | string | no       |
| bankName           | Specifies the name of the bank whose IO pin count is required. | string | no       |

## Examples

- `GetBankIOPinCount U3 1`
- `GetBankIOPinCount U10 Bank1`

## Related Commands

- [GetAvailableIOPinCount](#)
- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinCount](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

## GetBankIOPinNumberList

Returns a list of IO pin numbers available in the specified bank of the device instance.

## Return

string\_list

## Syntax

```
GetBankIOPinNumberList deviceInstanceName bankName
```

## Parameters

| Parameter          | Description                                                           | Type   | Optional |
|--------------------|-----------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance.                            | string | false    |
| bankName           | Specifies the name of the bank whose IO pin numbers list is required. | string | false    |

## Examples

- `GetBankIOPinNumberList U3 1`
- `GetBankIOPinNumberList U10 Bank1`

## Related Commands

- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetIOPinCount](#)
- [GetIOPinNumberList](#)

# GetBankName

Returns the name of the device instance bank in which the specified pin exists.

## Return

string

## Syntax

```
GetBankName device_instance_name pin_number
```

## Parameters

| Parameter            | Description                                                 | Type   | Optional |
|----------------------|-------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                  | string | false    |
| pin_number           | Specifies the pin number that belongs to the required bank. | string | false    |

## Examples

- `GetBankName U3 K16`
- `GetBankName U1 A1`

## Related Commands

[GetBanksNameList](#)

# GetBanksNameList

Returns a list of bank names of the specified device instance.

## Return

string\_list

## Syntax

```
GetBanksNameList deviceInstanceName
```

## Parameters

| Parameter          | Description                                                              | Type   | Optional |
|--------------------|--------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance whose bank names are required. | string | false    |

## Examples

- `GetBanksNameList U3`
- `GetBanksNameList U1`

## Related Commands

[GetBankName](#)

# GetBankVCCOlevel

Returns the VCCO voltage value of the specified device instance bank.

## Return

string

## Syntax

```
GetBankVCCOlevel device_instance_name bank_name
```

## Parameters

| Parameter            | Description                                                                 | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                  | string | false    |
| bank_name            | Specifies the name of the bank of which the VCCO voltage value is required. | string | false    |

## Examples

- `GetBankVCCOlevel U3 1`
- `GetBankVCCOlevel U1 Bank1`

## Related Commands

[GetBankVREFlevel](#)

# GetBankVREFlevel

Returns the VREF voltage value of the specified device instance bank.

## Return

string

## Syntax

```
GetBankVREFlevel device_instance_name bank_name
```

## Parameters

| Parameter            | Description                                                                 | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                  | string | false    |
| bank_name            | Specifies the name of the bank of which the VREF voltage value is required. | string | false    |

## Examples

- `GetBankVREFlevel U1 bank1`
- `GetBankVREFlevel U2 B2`

## Related Commands

[GetBankVCCOlevel](#)

# GetBoardDimensionUnits

Returns the board units.

## Return

string

## Syntax

```
GetBoardDimensionUnits
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`GetBoardDimensionUnits`

## Related Commands

[SetBoardDimensionUnits](#)

# GetBoardHeight

Returns the board height in design units.

## Return

double

## Syntax

`GetBoardHeight`

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`GetBoardHeight`

## Related Commands

- [GetBoardWidth](#)
- [SetBoardHeight](#)
- [SetBoardWidth](#)
- [GetBoardDimensionUnits](#)

# GetBoardWidth

Returns the board width in design units.

## Return

double

## Syntax

`GetBoardWidth`



## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetBoardWidth
```

## Related Commands

- [GetBoardHeight](#)
- [SetBoardHeight](#)
- [SetBoardWidth](#)
- [GetBoardDimensionUnits](#)

## GetCaptureINIFilePath

Returns the path of the capture.ini file used by the current design. The capture.ini file contains the settings required during generating OrCAD schematic.

## Return

string

## Syntax

```
GetCaptureINIFilePath
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetCaptureINIFilePath
```

## Related Commands

[SetCaptureINIFilePath](#)

## GetClockBuffer

Returns the clock buffer type that are defined for the specified clock region groups.

## Return

string

## Syntax

```
GetClockBuffer interfaceInstanceName groupName
```

## Parameters

| Parameter     | Description                          | Type   | Optional |
|---------------|--------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface. | string | false    |
| groupName     | Specifies the name of the group.     | string | false    |

## Examples

```
GetClockBuffer U2 clock_group
```

## Related Commands

[SetClockBuffer](#)

# GetConnectedInstancesList

Returns a list of instances connected to the specified synthesized net. Command does not work for power connection, fixed intern and fixed extern nets.

## Return

string\_list

## Syntax

```
GetConnectedInstancesList netName
```

## Parameters

| Parameter | Description                                                         | Type   | Optional |
|-----------|---------------------------------------------------------------------|--------|----------|
| netName   | Specifies the name of the net to which the instances are connected. | string | false    |

## Examples

- `GetConnectedInstancesList jtag_chain0_TCK`
- `GetConnectedInstancesList XP2_DDR2_PAR_IN`

## Related Commands

[GetNetName](#)

# GetConnectedNetGroupNames

Returns a list of NetGroups that are connected in the design.

## Return

string\_list

## Syntax

GetConnectedNetGroupName

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

GetConnectedNetGroupName

## Related Commands

[GetAvailableNetGroupName](#)

# GetConnectedPinCount

Returns the count of the number of pins connected for the specified device instance.

## Return

int

## Syntax

GetConnectedPinCount deviceInstanceName

## Parameters

| Parameter          | Description                                | Type   | Optional |
|--------------------|--------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance. | string | false    |

## Examples

- GetConnectedPinCount U3
- GetConnectedPinCount U1

## Related Commands

- [GetPinNameList](#)
- [GetPinNumber](#)
- [GetPinNumberList](#)
- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)

# GetContiguousSignal

Returns a list of contiguous signals that are set to the specified interface group.

## Return

string\_list

## Syntax

```
GetContiguousSignal instance_name groupName
```

## Parameters

| Parameter     | Description                                   | Type   | Optional |
|---------------|-----------------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface instance. | string | false    |
| groupName     | Specifies the name of the group.              | string | false    |

## Examples

```
GetContiguousSignal U2 Address_control
```

## Related Commands

- [GetAllContiguousSignal](#)
- [SetContiguousSignal](#)

# GetCurrentLicenseType

Returns the string of the current license used by FSP.

## Return

string

## Syntax

```
GetCurrentLicenseType
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetCurrentLicenseType
```

## Related Commands

- [GetAvailableLicenses](#)
- [GetAvaliableLicenseType](#)
- [SetLicenseType](#)

## GetDeepNWideGroupInstanceList

Returns a list of instances associated with the specified Deep and Wide group.

### Return

string\_list

### Syntax

```
GetDeepNWideGroupInstanceList dnw_group_name
```

### Parameters

| Parameter      | Description                                                                                  | Type   | Optional |
|----------------|----------------------------------------------------------------------------------------------|--------|----------|
| dnw_group_name | Specifies the name of the Deep and Wide group to which the required instance are associated. | string | false    |

### Examples

```
GetDeepNWideGroupInstanceList CommonGroup1
```

### Related Commands

- [CreateCommonGroup](#)
- [AddSecondary](#)
- [CreateWideBus](#)

## GetDeepNWideGroups

Returns a list of Deep and Wide groups present in the design.

### Return

string\_list

### Syntax

```
GetDeepNWideGroups
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetDeepNWideGroups
```

## Related Commands

- [GetDeepNWideGroupInstanceList](#)
- [CreateCommonGroup](#)
- [AddSecondary](#)
- [CreateWideBus](#)

## GetDehdlFPGAHierBlockLibrayName

Returns the library name where FPGA hierarchal blocks is to be generated.

### Return

string

### Syntax

```
GetDehdlFPGAHierBlockLibrayName
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetDehdlFPGAHierBlockLibrayName
```

## Related Commands

[SetDehdlFPGAHierBlockLibrayName](#)

## GetDesignDecapLibrarySymbolNameList

Returns the names of the libraries and symbols that are mapped to the decaps in the design.

### Return

string\_list

### Syntax

```
GetDesignDecapLibrarySymbolNameList
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetDesignDecapLibrarySymbolNameList
```

## Related Commands

- [AddDecap](#)
- [AddDecapOrCAD](#)
- [ChangeGlobalOrCADLibraryPath](#)

## GetDesignTopBlockLibAndName

Returns the name of the schematics root library and block.

### Return

string

### Syntax

```
GetDesignTopBlockLibAndName
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetDesignTopBlockLibAndName
```

## Related Commands

- [SetFPGAHierBlockLibAndName](#)
- [GetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)

## GetDeviceFamilyName

Returns the device family name for the specified device instance.

### Return

string

### Syntax

```
GetDeviceFamilyName device_name
```

### Parameters

| Parameter   | Description                                                     | Type   | Optional |
|-------------|-----------------------------------------------------------------|--------|----------|
| device_name | Specifies the name of the device whose family name is required. | string | false    |

## Examples

```
GetDeviceFamilyName U1
```

## Related Commands

- [GetSupportedFamilyNames](#)
- [GetTargetFPGAFamilies](#)
- [AddPart](#)
- [AddPartOrCAD](#)
- [PlaceInstance](#)
- [AddPartModel](#)

## GetDeviceInstanceList

Returns a list of device instance names that are present on the canvas.

### Return

string\_list

### Syntax

```
GetDeviceInstanceList
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetDeviceInstanceList
```

## Related Commands

- [GetInstanceNameList](#)
- [GetInterfaceInstanceList](#)
- [GetProtocolNames](#)

## GetDiffPairPin

Returns the other differential pair of the specified device pin number.

### Return

string

### Syntax

```
GetDiffPairPin device_name pin_number
```



## Parameters

| Parameter   | Description                                                          | Type   | Optional |
|-------------|----------------------------------------------------------------------|--------|----------|
| device_name | Specifies the name of the device instance.                           | string | false    |
| pin_number  | Specifies the pin number of which the differential pair is required. | string | false    |

## Examples

```
GetDiffPairPin U1 CLK_P
```

## Related Commands

[GetMGTPairPin](#)

# GetDoNotConnect

Return a list of do not connect pins of the specified device's bank.

## Return

string\_list

## Syntax

```
GetDoNotConnect device_instance_name bank_name
```

## Parameters

| Parameter            | Description                                                                   | Type   | Optional |
|----------------------|-------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device.                                             | string | false    |
| bank_name            | Specifies the name of the bank of which the do not connect pins are required. | string | false    |

## Examples

```
GetDoNotConnect U1 1
```

## Related Commands

- [GetAllDoNotConnect](#)
- [ResetAllDoNotConnect](#)
- [ResetDoNotConnectPin](#)
- [SetDoNotConnectPin](#)

# GetDontUseBanks

Returns a list of don't use banks that are set for the specified interface's group.

## Return

string\_list

## Syntax

```
GetDontUseBanks interface_name group_name
```

## Parameters

| Parameter      | Description                          | Type   | Optional |
|----------------|--------------------------------------|--------|----------|
| interface_name | Specifies the name of the interface. | string | false    |
| group_name     | Specifies the name of the group.     | string | false    |

## Examples

```
GetDontUseBanks XP1 Address_control
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

# GetDontUseBanksForProtocol

Returns a list of don't use banks for the specified device protocol group.

## Return

string\_list

## Syntax

```
GetDontUseBanksForProtocol protocol_name device_name group_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| protocol_name | Specifies the name of the protocol or interface protocol. | string | false    |
| device_name   | Specifies the name of the device.                         | string | false    |
| group_name    | Specifies the name of the protocol group.                 | string | false    |

## Examples

```
GetDontUseBanksForProtocol U2_U1 U1 Data_Input
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)
- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ReOptimizeProtocol](#)

## GetDraDirectoriesPaths

Returns a list of DRA directories paths. Generally dra paths info is being fetch from psmpath.

## Return

string\_list

## Syntax

```
GetDraDirectoriesPaths
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetDraDirectoriesPaths
```

## Related Commands

- [IsDraExist](#)
- [GetDraPath](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## GetDraPath

Returns the complete file path for a given DRA.

### Return

string

### Syntax

```
GetDraPath dra_name
```

### Parameters

| Parameter | Description                                                    | Type   | Optional |
|-----------|----------------------------------------------------------------|--------|----------|
| dra_name  | Specifies the name of the DRA file whose dra path is required. | string | false    |

### Examples

- `GetDraPath ff1517`
- `GetDraPath bga532.dra`

### Related Commands

- [IsDraExist](#)
- [GetDraDirectoriesPaths](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## GetEnvVariable

Returns the value of the specified environment variable such as CDSROOT,MY\_ENV etc and more.

### Return

string

### Syntax

```
GetEnvVariable envVarName
```

### Parameters

| Parameter  | Description                                                                | Type   | Optional |
|------------|----------------------------------------------------------------------------|--------|----------|
| envVarName | Specifies the name of the environment variable of which value is required. | string | false    |

## Examples

```
GetEnvVariable CDSROOT
```

## Related Commands

- [GetEnvVariables](#)
- [SetEnvVariable](#)

# GetEnvVariables

Returns a list of environment variables. This list includes both FSP and system variables.

## Return

string\_list

## Syntax

```
GetEnvVariables
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetEnvVariables
```

## Related Commands

[GetEnvVariable](#)

# GetFESymbolMapping

Returns the schematics part details of the specified instance such as library name, symbol name, and PTF data.

## Return

string

## Syntax

```
GetFESymbolMapping instance_name
```

## Parameters

| Parameter     | Description                                                              | Type   | Optional |
|---------------|--------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose schematic details are required. | string | false    |

## Examples

- `GetFESymbolMapping U1`
- `GetFESymbolMapping XP2`

## Related Commands

- [IsUsePart](#)
- [GetSymbolLibraryName](#)
- [GetSymbolPartName](#)
- [GetGenerateSymbolLibraryName](#)
- [GenerateAllegroSymbol](#)
- [GenerateOrCADSymbol](#)

## GetFPGAHierBlockLibAndName

Returns the name of the schematics FSP design block library name and block name.

### Return

string

### Syntax

```
GetFPGAHierBlockLibAndName instance_name
```

### Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

## Examples

```
GetFPGAHierBlockLibAndName U5
```

## Related Commands

- [SetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [GetDesignTopBlockLibAndName](#)

## GetFPGAPinMappingDirectoriesPaths

Returns a list of fpga pin mapping directories paths. Generally fpga pin mapping paths info is being fetch from Irfpath.

### Return

string\_list

## Syntax

```
GetFPGAPinMappingDirectoriesPaths
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetFPGAPinMappingDirectoriesPaths
```

## Related Commands

- [IsFPGAPinMappingFileExist](#)
- [GetFPGAPinMappingPath](#)
- [ReportAllFPGAPinMappingFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## GetFPGAPinMappingPath

Returns the complete file path for a given fpga pin mapping file.

## Return

string

## Syntax

```
GetFPGAPinMappingPath fpga_pin_mapping_file_name
```

## Parameters

| Parameter                  | Description                                                             | Type   | Optional |
|----------------------------|-------------------------------------------------------------------------|--------|----------|
| fpga_pin_mapping_file_name | Specifies the name of the fpga pin mapping file whose path is required. | string | false    |

## Examples

```
GetFPGAPinMappingPath ep4cgx150df27
```

## Related Commands

- [IsFPGAPinMappingFileExist](#)
- [GetFPGAPinMappingDirectoriesPaths](#)
- [ReportAllFPGAPinMappingFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## GetFSPBlockLibAndName

Returns the schematics FSP design block library name and the block name.

### Return

string

### Syntax

```
GetFSPBlockLibAndName
```

### Parameters

No Parameters

### Examples

```
GetFSPBlockLibAndName
```

## Related Commands

- [SetFPGAHierBlockLibAndName](#)
- [GetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [SetFSPBlockLibAndName](#)
- [GetDesignTopBlockLibAndName](#)

## GetFSPPartName

Returns the equivalent FSP FPGA part name name for the specified vendor's FPGA part name.

### Return

string

### Syntax

```
GetFSPPartName vendor_part_name
```



## Parameters

| Parameter        | Description                                                                           | Type   | Optional |
|------------------|---------------------------------------------------------------------------------------|--------|----------|
| vendor_part_name | Specifies the name of the vendor's part of which FSP's defined part name is required. | string | false    |

## Examples

- `GetFSPPartName XC4VLX25-10FFG668CS2`
- `GetFSPPartName XC6VLX240T-L1FF1156C`
- `GetFSPPartName EP4SGX230KF40C2ES`

## Related Commands

[GetSupportedFamilyNames](#)

# GetGenerateSymbolLibraryName

Returns name of the schematics symbol generation library.

## Return

string

## Syntax

```
GetGenerateSymbolLibraryName
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetGenerateSymbolLibraryName
```

## Related Commands

[SetGenerateSymbolLibraryName](#)

# GetGroupNameList

Returns a list of group names of the specified interface instance.

## Return

string\_list

## Syntax

```
GetGroupNameList interfaceInstance
```

## Parameters

| Parameter         | Description                                                                  | Type   | Optional |
|-------------------|------------------------------------------------------------------------------|--------|----------|
| interfaceInstance | Specifies the name of the interface instance whose group names are required. | string | false    |

## Examples

- `GetGroupNameList XP1`
- `GetGroupNameList U1`

## Related Commands

[GetTargetDevice](#)

# GetGroupOrBankPinNameList

Returns a list of pin names available in the specified group or bank of the specified instance.

## Return

string\_list

## Syntax

`GetGroupOrBankPinNameList instance_name group_bank_name`

## Parameters

| Parameter       | Description                                                    | Type   | Optional |
|-----------------|----------------------------------------------------------------|--------|----------|
| instance_name   | Name of the instance.                                          | string | false    |
| group_bank_name | Name of the group or bank of which the pin names are required. | string | false    |

## Examples

- `GetGroupOrBankPinNameList XP1 group1`
- `GetGroupOrBankPinNameList U2 bank1`

## Related Commands

[GetGroupOrBankPinNumberList](#)

# GetGroupOrBankPinNumberList

Returns a list of pin numbers of specified group or bank of the specified instance.

## Return

string\_list

## Syntax

```
GetGroupOrBankPinNumberList instance_name group_bank_name
```

## Parameters

| Parameter       | Description                                                      | Type   | Optional |
|-----------------|------------------------------------------------------------------|--------|----------|
| instance_name   | Name of the instance.                                            | string | false    |
| group_bank_name | Name of the group or bank of which the pin numbers are required. | string | false    |

## Examples

- `GetGroupOrBankPinNumberList XP1 group1`
- `GetGroupOrBankPinNumberList U2 bank1`

## Related Commands

[GetGroupOrBankPinNameList](#)

## GetInstanceCompletePartName

Returns the part and model name of the specified instance.

## Return

string

## Syntax

```
GetInstanceCompletePartName instanceName
```

## Parameters

| Parameter    | Description                                                                | Type   | Optional |
|--------------|----------------------------------------------------------------------------|--------|----------|
| instanceName | Specifies the name of the instance whose part and model names is required. | string | false    |

## Examples

```
#####examples ???#####
```

## Related Commands

- [GetInstancePartName](#)
- [GetPartWidth](#)
- [GetPartHeight](#)

## GetInstanceCustomAttributeValue

Returns the value of the custom attribute for the specified instance.

## Return

string

## Syntax

```
GetInstanceCustomAttributeValue instance_name custom_attrib_name
```

## Parameters

| Parameter          | Description                                                                         | Type   | Optional |
|--------------------|-------------------------------------------------------------------------------------|--------|----------|
| instance_name      | Specifies the name of the instance of which the custom attribute value is required. | string | false    |
| custom_attrib_name | Specifies the name of the custom attribute of which the value is required.          | string | false    |

## Examples

```
GetInstanceCustomAttributeValue U1 part_description
```

## Related Commands

- [DeleteInstanceCustomAttribute](#)
- [DeleteAllInstanceCustomAttributes](#)
- [AddInstanceCustomAttribute](#)

# GetInstanceDRAAbsoluteFilePath

Returns the absolute path to the DRA file path for the specified instance.

## Return

string

## Syntax

```
GetInstanceDRAAbsoluteFilePath instance_name
```

## Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

## Examples

```
GetInstanceDRAAbsoluteFilePath U1
```

## Related Commands

- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)
- [UpdateInstanceFootprint](#)
- [GetDraPath](#)
- [IsDraExist](#)
- [GetDraDirectoriesPaths](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)

## GetInstanceFootprint

Returns footprint information for given instance.

### Return

string

### Syntax

```
GetInstanceFootprint instance_name
```

## Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

## Examples

- `GetInstanceFootprint U1`
- `GetInstanceFootprint XP2`

## Related Commands

- [UpdateInstanceFootprint](#)
- [GetDraPath](#)
- [IsDraExist](#)
- [GetDraDirectoriesPaths](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)

## GetInstanceHeight

Returns the instance height in design unit for the specified instance.

### Return

double

## Syntax

```
GetInstanceHeight instance_name
```

## Parameters

| Parameter     | Description                                           | Type   | Optional |
|---------------|-------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the height is required. | string | false    |

## Examples

```
GetInstanceHeight XP1
```

## Related Commands

- [GetInstanceWidth](#)
- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

## GetInstanceMappingAbsolutePath

Returns the absolute path to the mapping file for the specified instance. This command will work only for non-FPGA instances that is linked to schematic symbol.

## Return

string

## Syntax

```
GetInstanceMappingAbsolutePath instance_name
```

## Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

## Examples

```
GetInstanceMappingAbsolutePath U1
```

## Related Commands

- [GetInstanceNameList](#)
- [GetInstanceRulesAbsolutePath](#)
- [GetRulesFilePaths](#)
- [GetResolvedRulesFilePaths](#)

## GetInstanceNameList

Returns a list of devices and interfaces instance names that are placed on the canvas.

### Return

string\_list

### Syntax

```
GetInstanceNameList
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetInstanceNameList
```

## Related Commands

- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)

## GetInstanceNamesOfJTAGChain

Returns the name of the instance involved in the specified JTAG chain.

### Return

string\_list

### Syntax

```
GetInstanceNamesOfJTAGChain jtag_chain_name
```

### Parameters

| Parameter       | Description                                                                         | Type   | Optional |
|-----------------|-------------------------------------------------------------------------------------|--------|----------|
| jtag_chain_name | Specifies the name of the JTAG chain to which the required instances are connected. | string | false    |

## Examples

```
GetInstanceNamesOfJTAGChain jtag1
```

## Related Commands

- [GetJTAGChains](#)
- [DeleteJTAGChain](#)
- [DefineJTAGChain](#)

## GetInstancePartName

Returns the part name of the specified instance.

### Return

string

### Syntax

```
GetInstancePartName instance_name
```

## Parameters

| Parameter     | Description                                                      | Type   | Optional |
|---------------|------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the instance name for which the part name is required. | string | false    |

## Examples

- `GetInstancePartName U3`
- `GetInstancePartName XP2`

## Related Commands

- [GetPartWidth](#)
- [GetPartHeight](#)

## GetInstancePinCustomAttributeValue

Returns the value of the custom attribute for the specified instance pin.

### Return

string

### Syntax

```
GetInstancePinCustomAttributeValue instance_name pin_number custom_attrib_name
```



## Parameters

| Parameter          | Description                                                                | Type   | Optional |
|--------------------|----------------------------------------------------------------------------|--------|----------|
| instance_name      | Specifies the name of the instance.                                        | string | false    |
| pin_number         | Specifies the pin number of the custom attribute value is required.        | string | false    |
| custom_attrib_name | Specified the name of the custom attribute of which the value is required. | string | false    |

## Examples

```
GetInstancePinCustomAttributeValue U1 A18 pin_type
```

## Related Commands

- [DeleteInstancePinCustomAttribute](#)
- [DeleteAllInstancePinCustomAttributes](#)
- [AddInstancePinCustomAttribute](#)

## GetInstanceRotation

Returns the rotation angle degree of the specified instance.

### Return

double

### Syntax

```
GetInstanceRotation instance_name
```

## Parameters

| Parameter     | Description                                                                        | Type   | Optional |
|---------------|------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the rotation angle degree is required. | string | false    |

## Examples

- `GetInstanceRotation U3`
- `GetInstanceRotation XP2`

## Related Commands

- [RotateInstance](#)
- [UpdateInstanceLocation](#)
- [InitInstancePinLocation](#)

## GetInstanceRulesAbsolutePath

Returns the absolute path of instance rules file.

## Return

string

## Syntax

```
GetInstanceRulesAbsolutePath instance_name
```

## Parameters

| Parameter     | Description                                                                  | Type   | Optional |
|---------------|------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the rules file path is required. | string | false    |

## Examples

```
GetInstanceRulesAbsolutePath U1
```

## Related Commands

- [GetInstanceNameList](#)
- [GetRulesFilePaths](#)
- [GetResolvedRulesFilePaths](#)
- [GetInstanceMappingAbsolutePath](#)

## GetInstanceSide

Returns the flip status of the specified instance. Valid values are top and bottom.

## Return

string

## Syntax

```
GetInstanceSide instance_name
```

## Parameters

| Parameter     | Description                                                              | Type   | Optional |
|---------------|--------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the flip status is required. | string | false    |

## Examples

- `GetInstanceSide U3`
- `GetInstanceSide XP2`

## Related Commands

- [FlipInstance](#)
- [RotateInstance](#)

## GetInstanceWidth

Returns the instance width in design unit for the specified instance.

### Return

double

### Syntax

```
GetInstanceWidth instance_name
```

### Parameters

| Parameter     | Description                                          | Type   | Optional |
|---------------|------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the width is required. | string | false    |

### Examples

```
GetInstanceWidth XP1
```

### Related Commands

- [GetInstanceHeight](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

## GetInstanceXCoordinate

Returns the center x co-ordinates in design unit for the specified instance.

### Return

double

### Syntax

```
GetInstanceXCoordinate instance_name
```

### Parameters

| Parameter     | Description                                                   | Type   | Optional |
|---------------|---------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the x co-ordinates is required. | string | false    |

## Examples

- `GetInstanceXCoordinate XP2`
- `GetInstanceXCoordinate U4`

## Related Commands

- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetInstanceHeight](#)
- [GetInstanceWidth](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

# GetInstanceYCoordinate

Returns the center y co-ordinates in design unit for the specified instance.

## Return

double

## Syntax

```
GetInstanceYCoordinate instance_name
```

## Parameters

| Parameter     | Description                                                          | Type   | Optional |
|---------------|----------------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the center y co-ordinates is required. | string | false    |

## Examples

```
GetInstanceYCoordinate U4
```

## Related Commands

- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceHeight](#)
- [GetInstanceWidth](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

## GetInstanceZOrderValue

Returns the Z order for the specified instance. Z order decides the display of the overlapping components in the Canvas.

### Return

double

### Syntax

```
GetInstanceZOrderValue instance_name
```

### Parameters

| Parameter     | Description                                                          | Type   | Optional |
|---------------|----------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the Z order is required. | string | false    |

### Examples

- `GetInstanceZOrderValue U4`
- `GetInstanceZOrderValue XP2`

### Related Commands

- [GetInstanceSide](#)
- [GetInstanceRotation](#)
- [FlipInstance](#)
- [RotateInstance](#)

## GetInterfaceInstanceList

Returns a list of interface instance names that are present on the canvas.

### Return

string\_list

### Syntax

```
GetInterfaceInstanceList
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetInterfaceInstanceList
```

## Related Commands

- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetProtocolNames](#)

## GetIOPinCount

Returns the number of IO pins count present in the specified device instance.

### Return

int

### Syntax

```
GetIOPinCount deviceInstanceName
```

### Parameters

| Parameter          | Description                                                                | Type   | Optional |
|--------------------|----------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance whose IO pins count is required. | string | false    |

### Examples

- `GetIOPinCount U1`
- `GetIOPinCount U2`

## Related Commands

- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinNumberList](#)

## GetIOPinNumberList

Returns a list of IO pin numbers for the specified device instance.

### Return

string\_list

### Syntax

```
GetIOPinNumberList deviceInstanceName
```

## Parameters

| Parameter          | Description                                                                  | Type   | Optional |
|--------------------|------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance whose IO pin numbers are required. | string | false    |

## Examples

- `GetIOPinNumberList U1`
- `GetIOPinNumberList U2`

## Related Commands

- [GetAvailableIOPinNumberList](#)
- [GetBankAvailableIOPinNumberList](#)
- [GetBankIOPinNumberList](#)
- [GetIOPinCount](#)

## GetIsolateStatus

Returns the Isolate High Speed Serial I/O's flag status for the specified device instance. This command is supported for Virtex4 or Virtex5 devices.

## Return

bool

## Syntax

```
GetIsolateStatus device_instance_name
```

## Parameters

| Parameter            | Description                                                                                            | Type   | Optional |
|----------------------|--------------------------------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance for which Isolate High Speed Serial IO's status is required. | string | false    |

## Examples

```
GetIsolateStatus U2
```

## Related Commands

[SetIsolateStatus](#)

## GetJedecType

Returns the footprint name or jedec type of the specified part name.

## Return

string

## Syntax

```
GetJedecType part_name
```

## Parameters

| Parameter | Description                                                                          | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------|--------|----------|
| part_name | Specifies the part name of the instances or rules file whose jedec type if required. | string | false    |

## Examples

```
GetJedecType 4vfx100ff1152
```

## Related Commands

- [GetDraPath](#)
- [GetDraDirectoriesPaths](#)
- [GetInstancePartName](#)

## GetJTAGChains

Returns a list of JTAG chain names present in the design.

## Return

string\_list

## Syntax

```
GetJTAGChains
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetJTAGChains
```

## Related Commands

- [DeleteJTAGChain](#)
- [DefineJTAGChain](#)

## GetMaximumNetGroupSize

Returns the value of the Maximum NetGroup size parameter. This value is considered by the auto-netgroup calls.



## Return

int

## Syntax

```
GetMaximumNetGroupSize
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetMaximumNetGroupSize
```

## Related Commands

- [SetMaximumNetGroupSize](#)
- [AutoNetGroupDesignGroupWise](#)

# GetMaxOutputsPerBank

Returns the maximum number of output pin count allowed for the specified bank of the device instance.

## Return

int

## Syntax

```
GetMaxOutputsPerBank device_instance_name bank_name
```

## Parameters

| Parameter            | Description                                                                       | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance name.                                   | string | false    |
| bank_name            | Specifies the name of the bank whose maximum number of outputs count is required. | string | false    |

## Examples

```
GetMaxOutputsPerBank U2 4
```

## Related Commands

[SetMaxOutputsPerBank](#)

# GetMGTPairPin

Returns a pair of the pin numbers of the specified MGT pin.

## Return

string

## Syntax

```
GetMGTPairPin device_instance_name pin_number
```

## Parameters

| Parameter            | Description                                                                  | Type   | Optional |
|----------------------|------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                   | string | false    |
| pin_number           | Specifies the pin number of the MGT pins for which pin numbers are required. | string | false    |

## Examples

```
GetMGTPairPin U1 A3
```

## Related Commands

[GetDiffPairPin](#)

# GetNetGroupSize

Returns the size of the specified NetGroup.

## Return

int

## Syntax

```
GetNetGroupSize net_group_name
```

## Parameters

| Parameter      | Description                                                | Type   | Optional |
|----------------|------------------------------------------------------------|--------|----------|
| net_group_name | Specifies the name of the NetGroup whose size is required. | string | false    |

## Examples

```
GetNetGroupSize NG1
```

## Related Commands

[SetMaximumNetGroupSize](#)

# GetNetName

Returns the net name that is connected to the specified instance pin.

## Return

string

## Syntax

```
GetNetName instance_name pin_number
```

## Parameters

| Parameter     | Description                                             | Type   | Optional |
|---------------|---------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                     | string | false    |
| pin_number    | Specifies the pin number to which the net is connected. | string | false    |

## Examples

- `GetNetName U10 B4`
- `GetNetName XP2 A1`

## Related Commands

- [GetConnectedInstancesList](#)
- [GetNetType](#)

# GetNetNamesOfNetGroup

Returns the net names of the specified NetGroup. This command returns null if no NetGroup is connected.

## Return

string\_list

## Syntax

```
GetNetNamesOfNetGroup net_group_name
```

## Parameters

| Parameter | Description                                                              | Type   | Optional |
|-----------|--------------------------------------------------------------------------|--------|----------|
| netGroup  | Specifies the name of the NetGroup for which the net names are required. | string | false    |

## Examples

```
GetNetNamesOfNetGroup NG1
```

## Related Commands

- [GetPinNumbersOfNetGroup](#)
- [AutoNetGroupDesignGroupWise](#)
- [SetMaximumNetGroupSize](#)

## GetNetType

Returns the net direction for the specified instance pin. Return values are Input, Output, and InOut.

### Return

string

### Syntax

```
GetNetType instance_name pinNumber
```

### Parameters

| Parameter     | Description                                                 | Type   | Optional |
|---------------|-------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                         | string | false    |
| pinNumber     | Specifies the pin number of which the net type is required. | string | false    |

### Examples

- `GetNetType U10 B4`
- `GeGetNetTypetNetName XP2 A1`

### Related Commands

- [GetConnectedInstancesList](#)
- [GetNetName](#)

## GetNoOfConnectedNetGroups

Returns the number of NetGroups that are connected in the design.

### Return

int

### Syntax

```
GetNoOfConnectedNetGroups
```

### Parameters

No Parameters

### Examples

```
GetNoOfConnectedNetGroups
```

## Related Commands

[GetNoOfNetGroups](#)

## GetNoOfNetGroups

Returns the number of NetGroups defined in the design.

### Return

int

### Syntax

```
GetNoOfNetGroups
```

### Parameters

No Parameters

### Examples

```
GetNoOfNetGroups
```

## Related Commands

[GetNoOfConnectedNetGroups](#)

## GetNotConnectedPinCount

Returns the count of the IO pins that are not connected for the specified device instance.

### Return

int

### Syntax

```
GetNotConnectedPinCount deviceInstanceName
```

### Parameters

| Parameter          | Description                                | Type   | Optional |
|--------------------|--------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance. | string | false    |

### Examples

- `GetNotConnectedPinCount U3`
- `GetNotConnectedPinCount U1`

## Related Commands

- [GetAvailableOPinNumberList](#)
- [GetBankOPinNumberList](#)
- [GetOPinCount](#)
- [GetOPinNumberList](#)

## GetOutputDirPath

Returns the absolute output directory path of the current design.

### Return

string

### Syntax

```
GetOutputDirPath
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetOutputDirPath
```

## Related Commands

[SetOutputDirPath](#)

## GetPartCustomAttributeValue

Returns the custom attribute value defined for the specified rules file.

### Return

string

### Syntax

```
GetPartCustomAttributeValue rulesName customAttributeName
```

### Parameters

| Parameter           | Description                                                                                                                    | Type   | Optional |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName           | Name of the rules file. Note: The rules file directory of the current design should be present in the rules file search paths. | string | false    |
| customAttributeName | Name of the custom attribute of which the value is required.                                                                   | string | false    |

## Examples

```
GetPartCustomAttributeValue mt46v64m4__sp6 CLASS
```

## Related Commands

[GetPartPinCustomAttribValue](#)

# GetPartDimensionUnit

Returns the unit of the specified part.

## Return

string

## Syntax

```
GetPartDimensionUnit rulesName
```

## Parameters

| Parameter | Description                                                                                                                                                        | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName | Name of the rules file of which the unit is required. The rules file directory must have either a FPGA model name or must be added in the rules file search paths. | string | false    |

## Examples

- `GetPartDimensionUnit mt46v64m4__sp6`
- `GetPartDimensionUnit cy7c1315bv18__v4__v5`
- `GetPartDimensionUnit 5vfx70tff1136`

## Related Commands

- [GetPartHeight](#)
- [GetPartWidth](#)

# GetPartHeight

Returns the part height in unit defined in the specified rules file.

## Return

double

## Syntax

```
GetPartHeight rulesName
```

## Parameters

| Parameter | Description                                                                                                                                                                | Type   | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName | Name of the rules file of which the height is required. Note: The rules file directory must have either a FPGA model name or must be added in the rules file search paths. | string | false    |

## Examples

- `GetPartHeight mt46v64m4__sp6`
- `GetPartHeight cy7c1315bv18__v4__v5`
- `GetPartHeight 5vfx70tff1136`

## Related Commands

- [GetPartDimensionUnit](#)
- [GetPartWidth](#)

# GetPartPinCustomAttribValue

Returns the custom attribute value defined for the specified pin or signal in the rules file definition.

## Return

string

## Syntax

```
GetPartPinCustomAttribValue rulesName pinNumber customAttributeName
```

## Parameters

| Parameter        | Description                                                                                                                                                                         | Type   | Optional |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName        | Specifies the name of the rules file. For this command to run successfully, the directory of the specified rules file must be added in the rules file paths for the current design. | string | false    |
| pinNumber        | Specifies the pin number of which the custom attributes is required.                                                                                                                | string | false    |
| customAttribName | Specifies the name of the custom attribute.                                                                                                                                         | string | false    |

## Examples

```
GetPartPinCustomAttribValue mt46v64m4__sp6 H2 PROP_X
```

## Related Commands

[GetPartCustomAttributeValue](#)

# GetPartPinXCoordinate

Returns part pin x location in unit defined in the specified rules file.



## Return

double

## Syntax

```
GetPartPinXCoordinate pinNumber rulesName
```

## Parameters

| Parameter | Description                                                                                                                                                                                    | Type   | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| pinNumber | Specifies the pin number.                                                                                                                                                                      | string | false    |
| rulesName | Name of the rules file name of which the X coordinate location is required. Note: The rules file directory must have either a FPGA model name or must be added in the rules file search paths. | string | false    |

## Examples

```
GetPartPinXCoordinate H2 mt46v64m4__sp6
```

## Related Commands

[GetPartPinYCoordinate](#)

# GetPartPinYCoordinate

Returns the Y coordinate location in unit for the specified part.

## Return

double

## Syntax

```
GetPartPinYCoordinate pinNumber rulesName
```

## Parameters

| Parameter | Description                                                                                                                                             | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| pinNumber | Specifies the pin number.                                                                                                                               | string | false    |
| rulesName | Name of the rules file of which the Y coordinate location is required. The rules file must be either a FPGA model name or from rules file search paths. | string | false    |

## Examples

```
GetPartPinYCoordinate H2 mt46v64m4__sp6
```

## Related Commands

[GetPartPinXCoordinate](#)

## GetPartWidth

Returns part width in unit defined in the specified rules file.

### Return

double

### Syntax

```
GetPartWidth rulesName
```

### Parameters

| Parameter | Description                                                                                                                         | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName | Name of the rules file of which width is required. The rules file must be either a FPGA model name or from rules file search paths. | string | false    |

### Examples

- `GetPartWidth mt46v64m4__sp6`
- `GetPartWidth cy7c1315bv18__v4__v5`
- `GetPartWidth 5vfx70tff1136`

### Related Commands

- [GetPartHeight](#)
- [GetPartDimensionUnit](#)
- [GetPartXOffset](#)
- [GetPartYOffset](#)

## GetPartXOffset

Returns part x offset in unit defined in the specified rules file.

### Return

double

### Syntax

```
GetPartXOffset rulesName
```

### Parameters

| Parameter | Description                                                                                                                                   | Type   | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName | Specifies the rules file name of which X offset is required. The rules file must be either a FPGA model name or from rules file search paths. | string | false    |

## Examples

- `GetPartXOffset mt46v64m4__sp6`
- `GetPartXOffset cy7c1315bv18__v4__v5`
- `GetPartXOffset 5vfx70tffl136`

## Related Commands

- [GetPartYOffset](#)
- [GetPartHeight](#)
- [GetPartWidth](#)
- [GetPartDimensionUnit](#)

## GetPartYOffset

Returns part y offset in unit defined in the specified rules file.

### Return

double

### Syntax

```
GetPartYOffset rulesName
```

### Parameters

| Parameter | Description                                                                                                                                   | Type   | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| rulesName | Specifies the rules file name of which Y offset is required. The rules file must be either a FPGA model name or from rules file search paths. | string | false    |

## Examples

- `GetPartYOffset mt46v64m4__sp6`
- `GetPartYOffset cy7c1315bv18__v4__v5`
- `GetPartYOffset 5vfx70tffl136`

## Related Commands

- [GetPartXOffset](#)
- [GetPartHeight](#)
- [GetPartWidth](#)
- [GetPartDimensionUnit](#)

## GetPCBOutlineHeight

Returns the pcb outline height for the current design. Returns 0 in case no custom pcb outline is set for the design.

## Return

double

## Syntax

```
GetPCBOutlineHeight
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetPCBOutlineHeight
```

## Related Commands

- [GetPCBOutlineWidth](#)
- [SetPCBOutlineHeight](#)
- [SetPCBOutlineWidth](#)
- [GetPCBOutlineSettings](#)
- [SetPCBOutlineSettings](#)

# GetPCBOutlineSettings

Returns the pcb outline settings string for the current design. Returns empty string in case no custom pcb outline is set for the design.

## Return

string

## Syntax

```
GetPCBOutlineSettings
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetPCBOutlineSettings
```

## Related Commands

- [GetPCBOutlineHeight](#)
- [GetPCBOutlineWidth](#)
- [SetPCBOutlineHeight](#)
- [SetPCBOutlineWidth](#)
- [SetPCBOutlineSettings](#)

## GetPCBOutlineWidth

Returns the pcb outline width for the current design. Returns zero in case no custom pcb outline is set for the design.

### Return

double

### Syntax

```
GetPCBOutlineWidth
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetPCBOutlineWidth
```

## Related Commands

- [GetPCBOutlineHeight](#)
- [SetPCBOutlineHeight](#)
- [SetPCBOutlineWidth](#)
- [GetPCBOutlineSettings](#)
- [SetPCBOutlineSettings](#)

## GetPinName

Returns the pin name of the specified instance pin number.

### Return

string

### Syntax

```
GetPinName instance_name pinNumber
```

## Parameters

| Parameter     | Description                                                      | Type   | Optional |
|---------------|------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                              | string | false    |
| pinNumber     | Name of the pin number of which the pin name is to be displayed. | string | false    |

## Examples

- `GetPinName U3 N8`
- `GetPinName U2 H1`

## Related Commands

- [GetConnectedPinCount](#)
- [GetPinNameList](#)
- [GetPinNumber](#)
- [GetPinNumberList](#)

# GetPinNameList

Returns a list of pin names of the specified instance.

## Return

string\_list

## Syntax

```
GetPinNameList instance_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the pin names are required. | string | false    |

## Examples

- `GetPinNameList U3`
- `GetPinNameList U2`

## Related Commands

- [GetConnectedPinCount](#)
- [GetPinName](#)
- [GetPinNumber](#)
- [GetPinNumberList](#)

## GetPinNumber

Returns the list of pin numbers for the specified instance. Generally, power and no connect pins can have more than one pin number for given pin name.

### Return

string\_list

### Syntax

```
GetPinNumber instance_name pinName
```

### Parameters

| Parameter     | Description                                          | Type   | Optional |
|---------------|------------------------------------------------------|--------|----------|
| instance_name | Name of the instance.                                | string | false    |
| pinName       | Name of the pin of which the pin number is required. | string | false    |

### Examples

- `GetPinNumber U3 IO_L1P_12`
- `GetPinNumber U2 QDRII_DLL_OFF`

### Related Commands

- [GetConnectedPinCount](#)
- [GetPinName](#)
- [GetPinNameList](#)
- [GetPinNumberList](#)

## GetPinNumberList

Returns a list of pin numbers available in the specified instance.

### Return

string\_list

### Syntax

```
GetPinNumberList instance_name
```

### Parameters

| Parameter     | Description                                                 | Type   | Optional |
|---------------|-------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which the pin numbers are required. | string | false    |

## Examples

- `GetPinNumberList U3`
- `GetPinNumberList U2`

## Related Commands

- [GetConnectedPinCount](#)
- [GetPinName](#)
- [GetPinNameList](#)
- [GetPinNumber](#)
- [GetPinNumberList](#)

## GetPinNumbersOfNetGroup

Returns the pin numbers of the specified NetGroup. For interface or virtual interface NetGroups, pins numbers are listed in syntax `interface_name.pin_number` and for device protocol NetGroups, `protocol_name.signal_name` is listed.

### Return

`string_list`

### Syntax

```
GetPinNumbersOfNetGroup net_group_name
```

### Parameters

| Parameter             | Description                                                                | Type   | Optional |
|-----------------------|----------------------------------------------------------------------------|--------|----------|
| <code>netGroup</code> | Specifies the name of the NetGroup for which the pin numbers are required. | string | false    |

## Examples

```
GetPinNumbersOfNetGroup NG1
```

## Related Commands

- [GetNetNamesOfNetGroup](#)
- [AutoNetGroupDesignGroupWise](#)
- [SetMaximumNetGroupSize](#)

## GetPinUseType

Returns the value as Positive or Negative for the specified pin. These two values determines the nature of the specified pin.

### Return

`string`



## Syntax

```
GetPinUseType device_instance_name pin_number
```

## Parameters

| Parameter            | Description                                                 | Type   | Optional |
|----------------------|-------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                  | string | false    |
| pin_number           | Specifies the pin number of which the pin type is required. | string | false    |

## Examples

```
GetPinUseType U1 B4
```

## Related Commands

[GetPinNumberList](#)

## GetPinXCoordinate

Returns the X co-ordinate of the specified pin in design unit.

## Return

double

## Syntax

```
GetPinXCoordinate pinNumber instance_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| pinNumber     | Specifies the pin number whose X co-ordinate is required. | string | false    |
| instance_name | Name of the instance.                                     | string | false    |

## Examples

- `GetPinXCoordinate G8 U3`
- `GetPinXCoordinate 168 XP2`

## Related Commands

- [GetPinYCoordinate](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetInstanceHeight](#)
- [GetInstanceWidth](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

## GetPinYCoordinate

Returns the Y co-ordinate of the specified pin in design unit.

### Return

double

### Syntax

```
GetPinYCoordinate pinNumber instance_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| pinNumber     | Specifies the pin number whose Y co-ordinate is required. | string | false    |
| instance_name | Name of the instance.                                     | string | false    |

## Examples

- `GetPinYCoordinate G8 U3`
- `GetPinYCoordinate 168 XP2`

## Related Commands

- [GetPinXCoordinate](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetInstanceHeight](#)
- [GetInstanceWidth](#)
- [GetPartPinXCoordinate](#)
- [GetPartPinYCoordinate](#)
- [GetBoardDimensionUnits](#)

## GetPowerPinsList

Returns a list of pin numbers of the power pins of the specified device instance bank.

## Return

string\_list

## Syntax

```
GetPowerPinsList device_instance_name {bank_name_list}
```

## Parameters

| Parameter            | Description                                                                     | Type        | Optional |
|----------------------|---------------------------------------------------------------------------------|-------------|----------|
| device_instance_name | Specifies the name of the device instance.                                      | string      | false    |
| bank_name_list       | Specifies the list of the bank names of which a list of power pins is required. | string_list | false    |

## Examples

- `GetPowerPinsList U3 [list 0 2]`
- `GetPowerPinsList U3 1`

## Related Commands

[GetBanksNameList](#)

# GetPowerRegulator

Returns the regulator name that is connected to the specified pin number.

## Return

string

## Syntax

```
GetPowerRegulator instance_name pinNumber
```

## Parameters

| Parameter     | Description                                                         | Type   | Optional |
|---------------|---------------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance.                                               | string | false    |
| pinNumber     | Specifies the pin number to which the power regulator is connected. | string | false    |

## Examples

- `GetPowerRegulator U3 G8`
- `GetPowerRegulator XP2 168`

## Related Commands

- [SetPowerRegulator](#)
- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## GetPowerRegulatorName

Returns the regulator name that is connected to the specified pin number.

### Return

string

### Syntax

```
GetPowerRegulatorName instance_name pinNumber
```

### Parameters

| Parameter     | Description                                                         | Type   | Optional |
|---------------|---------------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance.                                               | string | false    |
| pinNumber     | Specifies the pin number to which the power regulator is connected. | string | false    |

### Examples

- `GetPowerRegulatorName U3 G8`
- `GetPowerRegulatorName XP2 168`

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## GetPowerRegulators

Returns a list of power regulators defined in the design.

### Return

string\_list

### Syntax

```
GetPowerRegulators
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetPowerRegulators
```

### Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## GetPowerRegulatorVoltage

Returns the voltage value of the specified power regulator.

### Return

double

### Syntax

```
GetPowerRegulatorVoltage regulator_name
```

### Parameters

| Parameter      | Description                                                                 | Type   | Optional |
|----------------|-----------------------------------------------------------------------------|--------|----------|
| regulator_name | Name of the power regulator for which the voltage value is to be displayed. | string | false    |

## Examples

```
GetPowerRegulatorVoltage V_0_9
```

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [SetPowerRegulatorVoltage](#)

## GetPreservePins

Returns a list of pin numbers of the specified device bank that are preserved.

### Return

string\_list

### Syntax

```
GetPreservePins device_instance_name bank_name
```

## Parameters

| Parameter            | Description                                                                 | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                  | string | false    |
| bank_name            | Specifies the name of the bank of which a list of preserved pins is needed. | string | false    |

## Examples

```
GetPreservePins U1 3
```

## Related Commands

- [SetPreservePins](#)
- [SetDevicePreservePins](#)

## GetProjectFilePath

Returns the path of the current project directory.

## Return

string

## Syntax

```
GetProjectFilePath
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

- `GetProjectFilePath`
- `GetProjectFilePath`

## Related Commands

- [GetProjectName](#)
- [OpenProject](#)
- [CloseProject](#)
- [SaveProject](#)
- [SaveProjectAs](#)

# GetProjectName

Returns the name of the current project.

## Return

string

## Syntax

```
GetProjectName
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetProjectName
```

## Related Commands

- [GetProjectFilePath](#)
- [OpenProject](#)
- [CloseProject](#)
- [SaveProject](#)
- [SaveProjectAs](#)

## GetProtocolNames

Returns a list of protocols names used in the design.

### Return

string\_list

### Syntax

```
GetProtocolNames
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetProtocolNames
```

## Related Commands

- [RenameProtocol](#)
- [ChangeProtocolOrder](#)
- [ExportCSVFromProtocol](#)
- [ExportProtocolDefinition](#)
- [ReOptimizeProtocol](#)
- [UpdateProtocolFromCSV](#)
- [SetAutoNetGroupProtocol](#)
- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)

## GetRegulatorName

Returns a regulator name that is connected to the specified device instance pin.

### Return

string



## Syntax

```
GetRegulatorName device_instance_name pin_number
```

## Parameters

| Parameter            | Description                                                                | Type   | Optional |
|----------------------|----------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                 | string | no       |
| pin_number           | Specifies the pin number of a pin of which the regulator name is required. | string | no       |

## Examples

- `GetRegulatorName U3 G8`
- `GetRegulatorName U3 H1`

## Related Commands

- [AutoAddPowerRegulators](#)
- [AutoMapPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [GetPowerRegulatorName](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## GetReleasePath

Returns the installation directory path \$CDSROOT/tools/fsp of FSP.

## Return

string

## Syntax

```
GetReleasePath
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetReleasePath
```

## Related Commands

- [GetEnvVariable](#)
- [GetEnvVariables](#)
- [GetWorkingDir](#)

## GetResolvedRulesFilePaths

Returns a list of complete rules file paths that are set for the current design.

### Return

string\_list

### Syntax

```
GetResolvedRulesFilePaths
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetResolvedRulesFilePaths
```

## Related Commands

- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [RemoveRulesFilePath](#)
- [GetRulesWorkingDir](#)
- [SetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [CheckDesignConsistency](#)

## GetResources

Returns the attached resources information from the constraint files.

### Return

string

### Syntax

```
GetResources -d deviceInstanceName [-i interfaceInstanceName]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                         | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device instance.                                                                                                                                                                          | string | false    |
| -i        | Specifies the name of the interface instance or protocol of which the resource is required. In case argument is not specified, the command returns the resources that is defined for the specified device instance. | string | true     |

## Examples

```
GetResources -d U1 -i U2
```

## Related Commands

- [MapResources](#)

## GetRuleFilePath

Returns complete file path for given rules name.

## Return

string

## Syntax

```
GetRuleFilePath rulesName
```

## Parameters

| Parameter | Description                                                       | Type   | Optional |
|-----------|-------------------------------------------------------------------|--------|----------|
| rulesName | Specifies the name of the rules file whose file path is required. | string | false    |

## Examples

- `GetRuleFilePath ddr2_dimm_x4__v4__v5`
- `GetRuleFilePath mt46v64m4__sp6`

## Related Commands

- [GetRulesFilePaths](#)
- [RemoveRulesFilePath](#)
- [GetRulesWorkingDir](#)
- [SetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [GetResolvedRulesFilePaths](#)
- [CheckDesignConsistency](#)

## GetRulesFilePaths

Returns a list of rules file paths that were set for the current design. This command does not resolve the soft paths. To get the list of resolved rules file paths use the GetResolvedRulesFilePaths command.

### Return

string\_list

### Syntax

```
GetRulesFilePaths
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetRulesFilePaths
```

### Related Commands

- [GetRuleFilePath](#)
- [RemoveRulesFilePath](#)
- [GetRulesWorkingDir](#)
- [SetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [GetResolvedRulesFilePaths](#)
- [CheckDesignConsistency](#)

## GetRulesWorkingDir

Returns the rules working directory path that is set for the current design.

### Return

string

### Syntax

```
GetRulesWorkingDir
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetRulesWorkingDir
```

## Related Commands

- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [RemoveRulesFilePath](#)
- [SetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [GetResolvedRulesFilePaths](#)
- [CheckDesignConsistency](#)

## GetSchematicBoardFileName

Returns the name of the Allegro board file (.brd).

### Return

string

### Syntax

```
GetSchematicBoardFileName
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
GetSchematicBoardFileName
```

## Related Commands

- [SetSchematicBoardFilePath](#)
- [GetSchematicBoardFilePath](#)
- [SetSchematicBoardFileName](#)

## GetSchematicBoardFilePath

Returns the path of the directory where the Allegro board file (.brd) is generated.

### Return

string

### Syntax

```
GetSchematicBoardFilePath
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetSchematicBoardFilePath
```

## Related Commands

- [SetSchematicBoardFilePath](#)
- [GetSchematicBoardFileName](#)
- [SetSchematicBoardFileName](#)

# GetSchematicsEnvironment

Returns the schematic tool type of the design.

## Return

string

## Syntax

```
GetSchematicsEnvironment
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetSchematicsEnvironment
```

## Related Commands

[GetSchematicsEnvironment](#)

# GetSchematicsSymbolFileReference

Returns the path of the symbol directory for the Orcad symbol.

## Return

string

## Syntax

```
GetSchematicsSymbolFileReference instance_name
```

## Parameters

| Parameter     | Description                                                                   | Type   | Optional |
|---------------|-------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the symbol file path is required. | string | false    |

## Examples

```
GetSchematicsSymbolFileReference U1
```

## Related Commands

[ChangeSchematicsSymbolFileReference](#)

# GetSearchNReplaceNetNamePattern

Returns True if search and replace name pattern option is ON.

## Return

bool

## Syntax

```
GetSearchNReplaceNetNamePattern
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetSearchNReplaceNetNamePattern
```

## Related Commands

- [SetSearchNReplaceNetNamePattern](#)
- [AddSearchNReplaceNetNamePattern](#)
- [RemoveSearchNReplaceNetNamePatterns](#)

# GetSnapshotTimeInterval

Returns the snapshot time interval of the current design. The command saves a copy of the design on specified time interval.

## Return

int

## Syntax

```
GetSnapshotTimeInterval
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetSnapshotTimeInterval
```

## Related Commands

[SetSnapshotTimeInterval](#)

# GetSupportedFamilyNames

Returns a list of FPGA families that are supported by FSP.

## Return

string\_list

## Syntax

```
GetSupportedFamilyNames
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetSupportedFamilyNames
```

## Related Commands

- [GetTargetFPGAFamilies](#)
- [GetDeviceFamilyName](#)
- [AddPart](#)
- [AddPartOrCAD](#)
- [AddPartModel](#)
- [PlaceInstance](#)

# GetSwappablePins

Returns a list of swappable pin numbers for the specified device instance pin. The scope could be within a device, across all components or within a Bank.

## Return

string\_list



## Syntax

```
GetSwappablePins deviceInstanceName pinNumber [-bank] [-all]
```

## Parameters

| Parameter          | Description                                                          | Type   | Optional |
|--------------------|----------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specify the name of the device instance.                             | string | false    |
| pinNumber          | Specify pin number.                                                  | string | false    |
| -bank              | Indicates if the scope of pin swap if limited to only current Bank.  | bool   | true     |
| -all               | Indicates if the swappable pins of all the components are requested. | bool   | true     |

## Examples

- `GetSwappablePins U1 H1`
- `GetSwappablePins U1 H1 -bank`
- `GetSwappablePins U1 H1 -all`

## Related Commands

[GetDeviceInstanceList](#)

# GetSymbolLibraryName

Returns the library name of the schematics symbol of the specified instance.

## Return

string

## Syntax

```
GetSymbolLibraryName instance_name
```

## Parameters

| Parameter     | Description                                                                | Type   | Optional |
|---------------|----------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose symbols library name is required. | string | false    |

## Examples

- `GetSymbolLibraryName U1`
- `GetSymbolLibraryName XP1`

## Related Commands

- [IsUsePart](#)
- [GetFESymbolMapping](#)
- [GetSymbolPartName](#)
- [GetGenerateSymbolLibraryName](#)
- [GenerateAllegroSymbol](#)
- [GenerateOrCADSymbol](#)

## GetSymbolPartName

Returns the schematics symbol name of the specified instance.

### Return

string

### Syntax

```
GetSymbolPartName instance_name
```

### Parameters

| Parameter     | Description                                                                  | Type   | Optional |
|---------------|------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose schematics symbol name is required. | string | false    |

### Examples

```
GetSymbolPartName U2
```

## Related Commands

- [IsUsePart](#)
- [GetFESymbolMapping](#)
- [GetSymbolLibraryName](#)
- [GetGenerateSymbolLibraryName](#)
- [GenerateAllegroSymbol](#)
- [GenerateOrCADSymbol](#)

## GetTargetDevice

Returns the list of the device instance names to which the specified interface group is targeted to.

### Return

string\_list

### Syntax

```
GetTargetDevice instance_name group_name
```

## Parameters

| Parameter     | Description                                | Type   | Optional |
|---------------|--------------------------------------------|--------|----------|
| instance_name | Specifies name of the instance.            | string | false    |
| group_name    | Specifies the name of the interface group. | string | false    |

## Examples

- `GetTargetDevice XPl Data_Output`
- `GetTargetDevice U1 group1`

## Related Commands

- [GetAllTargetDevice](#)
- [GetTargetInstanceListForDevice](#)
- [TargetDevice](#)
- [TargetToMultipleDevices](#)
- [GetGroupNameList](#)

# GetTargetFPGAFamilies

Returns a list of FPGA families to which the specified rules file is targeted.

## Return

string\_list

## Syntax

```
GetTargetFPGAFamilies rule_sname
```

## Parameters

| Parameter  | Description                           | Type   | Optional |
|------------|---------------------------------------|--------|----------|
| rules_name | Specifies the name of the rules file. | string | false    |

## Examples

- `GetTargetFPGAFamilies pci_x64`
- `GetTargetFPGAFamilies ddr2_sdram_x16_sd_84bga_aiigx`

## Related Commands

- [GetSupportedFamilyNames](#)
- [GetDeviceFamilyName](#)
- [AddPart](#)
- [AddPartOrCAD](#)
- [AddPartModel](#)
- [PlaceInstance](#)

## GetTargetInstanceListForDevice

Returns a list of interface, virtual interface, and protocols that are connected to the specified device instance.

### Return

string\_list

### Syntax

```
GetTargetInstanceListForDevice deviceName
```

### Parameters

| Parameter  | Description                                | Type   | Optional |
|------------|--------------------------------------------|--------|----------|
| deviceName | Specifies the name of the device instance. | string | false    |

### Examples

- ```
GetTargetInstanceListForDevice U3
```
- ```
GetTargetInstanceListForDevice U4
```

## Related Commands

- [GetAllTargetDevice](#)
- [GetTargetDevice](#)
- [TargetDevice](#)
- [TargetToMultipleDevices](#)

## GetTerminationNames

Returns a list of names of the terminations and power filters defined in the design.

### Return

string\_list

### Syntax

```
GetTerminationNames
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetTerminationNames
```

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [SpecifyDiffPinTermination](#)

# GetTerminationSymbol

Returns the symbol information that is mapped to the specified termination.

## Return

string

## Syntax

```
GetTerminationSymbol termination_name
```

## Parameters

| Parameter        | Description                                                                   | Type   | Optional |
|------------------|-------------------------------------------------------------------------------|--------|----------|
| termination_name | Specifies the name of the termination to which the required symbol is mapped. | string | false    |

## Examples

```
GetTerminationSymbol ser_term
```

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)

## GetUseBanks

Returns a list of bank names to which the specified interface's or virtual interface's group is targeted.

### Return

string\_list

### Syntax

```
GetUseBanks instance_name groupName
```

### Parameters

| Parameter     | Description                          | Type   | Optional |
|---------------|--------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface. | string | false    |
| groupName     | Specifies the name of the group.     | string | false    |

### Examples

- `GetUseBanks XP2 Address_Control`
- `GetUseBanks XP2 Address_Control`

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

## GetUseBanksForProtocol

Returns the Use Bank for the specified device protocol.

### Return

string\_list

### Syntax

```
GetUseBanksForProtocol protocolName
```

### Parameters

| Parameter    | Description                                | Type   | Optional |
|--------------|--------------------------------------------|--------|----------|
| protocolName | Specifies the name of the device protocol. | string | false    |
| deviceName   | Specifies the name of the device.          | string | false    |
| groupName    | Specifies the name of the protocol group.  | string | false    |

### Examples

```
GetUseBanksForProtocol J1,U3 U3 Address_control
```

### Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)
- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ReOptimizeProtocol](#)

## GetVectorCloseBrace

Returns close brace character used in the current design. You can use the config.ini file located at \$cdsroot/share/cdssetup/fsp to update site specific vector notations for the design.

### Return

string

## Syntax

GetVectorCloseBrace

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

GetVectorCloseBrace

## Related Commands

[GetVectorOpenBrace](#)

# GetVectorOpenBrace

Returns open brace character used in the current design. You can use the config.ini file located at \$cdsroot/share/cdssetup/fsp to update site specific vector notations for the design.

## Return

string

## Syntax

GetVectorOpenBrace

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

GetVectorOpenBrace

## Related Commands

[GetVectorCloseBrace](#)

# GetVIInstanceList

Returns a list of virtual interfaces that are targeted to the specified device instance.

## Return

string\_list

## Syntax

GetVIInstanceList deviceInstName



## Parameters

| Parameter      | Description                                                                                      | Type   | Optional |
|----------------|--------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstName | Specifies the name of the device instance to which the required virtual interfaces are targeted. | string | no       |

## Examples

- `GetVIInstanceList U3`
- `GetVIInstanceList U4`

## Related Commands

- [GetDeviceInstanceList](#)
- [GetVirtualInterfaceNames](#)

# GetVirtualInterfaceNames

Returns a list of virtual interfaces names present in the design.

## Return

string\_list

## Syntax

```
GetVirtualInterfaceNames [deviceInstName]
```

## Parameters

| Parameter      | Description                                                                                      | Type   | Optional |
|----------------|--------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstName | Specifies the name of the device instance to which the required virtual interfaces are targeted. | string | true     |

## Examples

- `GetVirtualInterfaceNames`
- `GetVirtualInterfaceNames U3`
- `GetVirtualInterfaceNames U4`

## Related Commands

- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)
- [GetProtocolNames](#)

# GetVoltageLevel

Returns the voltage value of the specified device instance power pin. For this command to run successfully, you must enter pin number of a power pin.

## Return

string

## Syntax

```
GetVoltageLevel device_instance_name pin_number
```

## Parameters

| Parameter            | Description                                                                       | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                        | string | false    |
| pin_number           | Specifies the pin number of the power pin of which the voltage value is required. | string | false    |

## Examples

- `GetVoltageLevel U3 AK15`
- `GetVoltageLevel U3 AL12`

## Related Commands

- [AddPowerRegulator](#)
- [AutoAddPowerRegulators](#)

# GetWorkingDir

Returns the path of the FSP's working directory. By default, the projects are created and saved in the working directory.

## Return

string

## Syntax

```
GetWorkingDir
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
GetWorkingDir
```

## Related Commands

- [GetReleasePath](#)
- [GetEnvVariable](#)
- [GetEnvVariables](#)

## HideAllNets

Hides all nets on the canvas.

### Return

bool

### Syntax

```
HideAllNets
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
HideAllNets
```

### Related Commands

[HideInstanceNets](#)

## HideInstanceNets

Hides all nets which are connected to the specified instance on the canvas.

### Return

bool

### Syntax

```
HideInstanceNets instance_name
```

### Parameters

| Parameter     | Description                                            | Type   | Optional |
|---------------|--------------------------------------------------------|--------|----------|
| instance_name | Name of the instance of which nets that is to be hide. | string | false    |

### Examples

```
HideInstanceNets
```

### Related Commands

[HideAllNets](#)

## ImportConstraints

Modifies the interface instance or protocol signals connectivity by importing the constraint file.

### Return

bool

### Syntax

```
ImportConstraints -d device_instance_name -f constraint_file_path -i interface_or_protocol_name -r run_device
```

### Parameters

| Parameter | Description                                                                                                               | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device instance for which the connectivity is to be updated.                                    | string | false    |
| -f        | Specifies the constraint file path that is to be imported.                                                                | string | false    |
| -i        | Specifies the name of the interface or protocol whose signal's use pins is to be updated.                                 | string | false    |
| -r        | Specifies whether the connectivity is to be updated based on the modified use pins settings. Valid values are Yes and No. | string | false    |

### Examples

- `ImportConstraints -d U1 -f ./u1.ucf -i U2 -r yes`
- `ImportConstraints -d U1 -f ./u1.ucf -i U1_U3 -r no`

### Related Commands

- [GenerateConstraintFiles](#)
- [ExportConstraints](#)

## ImportCSVInDesignConnectivity

Imports the pin and connectivity information from the comma separated value (CSV) file in Design Connectivity.

### Return

bool

### Syntax

```
ImportCSVInDesignConnectivity -f csv_file_path -m column_mapping -r {reference_column_name_list} [-d delimiter] [-i ignore_row_numbers] [-a]
```

## Parameters

| Parameter | Description                                                                                                                                                                              | Type              | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -f        | Specifies path to the CSV file to be imported.                                                                                                                                           | string            | false    |
| -m        | Use this option to map the column name to column number.                                                                                                                                 | string_string_map | false    |
| -r        | Specifies the list of reference column names.                                                                                                                                            | string_list       | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), ' '(space).                                                              | string            | true     |
| -i        | Specifies a list of row numbers (in comma separated format) that is to be ignored while importing the CSV file. By default, the command does not ignore any rows.                        | string            | true     |
| -a        | Specifies whether to resets all the invalid values that are imported from the CSV file. In case this argument is not specified, the command fails due to invalid values in the CSV file. | bool              | true     |

## Examples

- ImportCSVInDesignConnectivity -f ./de\_export\_pin\_view.csv -d , -i 1 -r {"Pin Number\" \"Target Device\"} -m {"NetGroup\" 27 \"Pin Number\" 3 \"Instance/Protocol Name\" 37 \"Target Device\" 35 \"Connection Type\" 19} -a
- ImportCSVInDesignConnectivity -f ./de\_export\_pin\_view.csv -d , -i 1 -r {"FSP\_UID\"} -m {"NetGroup\" 27 \"FSP\_UID\" 29}
- ImportCSVInDesignConnectivity -f ./de\_export\_net\_view.csv -d , -i 1 -r {"Pin Number\"} -m {"NetGroup\" 27 \"Pin Number\" 3 \"Target Device\" 33 \"Instance/Protocol Name\" 35}
- ImportCSVInDesignConnectivity -f ./de\_export\_net\_view.csv -d , -i 1 -r {"FSP\_UID\"} -m {"NetGroup\" 27 \"FSP\_UID\" 29}
- ImportCSVInDesignConnectivity -f ./de\_export\_net\_view.csv -d , -i 1 -r {"Pin/Port Name\"} -m {"NetGroup\" 27 \"Pin Number\" 3 \"Target Device\" 33 \"Instance/Protocol Name\" 35 \"Pin/Port Name\" 2 \"Connection Type\" 19} -a

## Related Commands

- [SetDesignConnectivityView](#)
- [ExportCSVfromDesignConnectivity](#)

## ImportCSVInDesignExplorer

Imports the pin and connectivity information from the comma separated value (CSV) file in Design Connectivity.

## Return

bool

## Syntax

```
ImportCSVInDesignExplorer -f csv_file_path -d delimiter -m column_mapping -r {reference_column_name_list} [-i ignore_row_numbers] [-a]
```

## Parameters

| Parameter | Description                                                                                                                                                                              | Type              | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -f        | Specifies path to the CSV file to be imported.                                                                                                                                           | string            | no       |
| -d        | Specifies the delimiter used in the CSV file. For example,(comma), (pipe),;(semicolon),:(colon)                                                                                          | string            | no       |
| -m        | Use this option to map the column name to column number.                                                                                                                                 | string_string_map | no       |
| -r        | Specifies the list of reference column names.                                                                                                                                            | string_list       | no       |
| -i        | Specifies a list of row numbers, in comma separated format, to be ignored from the CSV file. By default, the command does not ignore any rows.                                           | string            | yes      |
| -a        | Specifies whether to resets all the invalid values that are imported from the CSV file. In case this argument is not specified, the command fails due to invalid values in the CSV file. | bool              | yes      |

## Examples

- ImportCSVInDesignExplorer -f ./de\_export\_pin\_view.csv -d , -i 1 -r {"Pin Number" "Target Device"} -m {"NetGroup" 27 "Pin Number" 3 "Instance/Protocol Name" 37 "Target Device" 35 "Connection Type" 19} -a
- ImportCSVInDesignExplorer -f ./de\_export\_pin\_view.csv -d , -i 1 -r {"FSP\_UID"} -m {"NetGroup" 27 "FSP\_UID" 29}
- ImportCSVInDesignExplorer -f ./de\_export\_net\_view.csv -d , -i 1 -r {"Pin Number"} -m {"NetGroup" 27 "Pin Number" 3 "Target Device" 33 "Instance/Protocol Name" 35}
- ImportCSVInDesignExplorer -f ./de\_export\_net\_view.csv -d , -i 1 -r {"FSP\_UID"} -m {"NetGroup" 27 "FSP\_UID" 29}
- ImportCSVInDesignExplorer -f ./de\_export\_net\_view.csv -d , -i 1 -r {"Pin/Port Name"} -m {"NetGroup" 27 "Pin Number" 3 "Target Device" 33 "Instance/Protocol Name" 35 "Pin/Port Name" 2 "Connection Type" 19} -a

## Related Commands

- [SetDesignExplorerView](#)
- [ExportCSVfromDesignExplorer](#)

## ImportDecaps

Import Decaps info.

## Return

bool

## Syntax

```
ImportDecaps [-instances {instance_name_list}] -file filename
```

## Parameters

| Parameter  | Description                                                                                                                                                   | Type        | Optional |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -instances | Decaps will be imported for the specified instances from specified file. If instance names are not specified then, decpas for all instances will be imported. | string_list | true     |
| -file      | Specifies the export file path.                                                                                                                               | string      | false    |

## Examples

- `ImportDecaps -instances [list U1 U2] -file ./decaps.xml`
- `ImportDecaps -instances U1 -file ./decaps.xml`
- `ImportDecaps -file ./decaps.xml`

## Related Commands

[ExportDecaps](#)

# ImportFromAllegroBoard

Creates a new design by importing the Allegro board file.

## Return

bool

## Syntax

```
ImportFromAllegroBoard -b allegro_board_file_path -s settings_file
```

## Parameters

| Parameter | Description                                                                                                  | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------|--------|----------|
| -b        | Specifies the board file path that is to be imported.                                                        | string | false    |
| -s        | Specifies the details of the components (rules file, mapping file, schematic symbol) that is to be imported. | string | false    |

## Examples

```
ImportFromAllegroBoard -b ./project20.brd -s ./project20_import_settings.xml
```

## Related Commands

- [GenerateDEHDL Schematics](#)
- [GenerateOrCAD Schematics](#)
- [ImportPlacementXMLFile](#)
- [GenerateLayoutData](#)
- [UpdateLayoutData](#)

# ImportInstanceConstraints

Imports the constraints from the external file into into the specified instance.

## Return

bool

## Parameters

| Parameter               | Description                                                                      | Type   | Optional |
|-------------------------|----------------------------------------------------------------------------------|--------|----------|
| instance_name           | Name of the instance.                                                            | string | no       |
| ucf_qsf_filename        | Specifies the name of the UCF or QSF file that is to be used for importing.      | string | no       |
| verilog_vhdl_filename   | Specifies the name of the verilog or vhdl file that is to be used for importing. | string | no       |
| verilog_vhdl_modulename | Name of the verilog or vhdl module.                                              | string | no       |

## Examples

No Examples

## Related Commands

- [GeneratePlanAheadScripts](#)
- [ExportDeviceConstraints](#)
- [GenerateConstraintFiles](#)
- [ImportConstraints](#)
- [GenerateVerilogBrdDescFile](#)

## ImportPDC

Modifies the interface instance or protocol signals connectivity by importing the PDC file.

## Return

bool

## Syntax

```
ImportPDC -d device_name -f file_path [-i {interface_list}] [-v]
```

## Parameters

| Parameter | Description                                                                               | Type        | Optional |
|-----------|-------------------------------------------------------------------------------------------|-------------|----------|
| -d        | Specifies the name of the device instance for which the connectivity is to be updated.    | string      | false    |
| -f        | Specifies the constraint file path that is to be imported.                                | string      | false    |
| -i        | Specifies the name of the interface or protocol whose signal's use pins is to be updated. | string_list | true     |
| -v        | Specifies whether the preserved VREFs to be imported from the specified file.             | bool        | true     |

## Examples

- `ImportPDC -d U1 -f ./U1.pdc -i [list U2 U4]`
- `ImportPDC -d U5 -f ./U5.pdc -i [list U1 U3] -v`



## Related Commands

- [ResetPDCPreservedVREFs](#)
- [ImportConstraints](#)

## ImportPinAssignmentsForConnector

Updates pin assignments of the specified connector with the values coming from the specified CSV file.

### Return

bool

### Syntax

```
ImportPinAssignmentsForConnector -n connector_name -c csv_file_path -m column_mapping -r reference_column [-d delimiter] [-i ignore_rows]
```

### Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -n        | Specifies the name of the connector that is to be updated.                                                                                                     | string            | false    |
| -c        | Specifies the path of the CSV file that is to be imported to update the connector pin properties.                                                              | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                    | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), ' '(space) etc...                              | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

### Examples

```
ImportPinAssignmentsForConnector -c ./update_connector_csv.csv -d , -i 1 -r \"Pin Number\" -m {\"Pin Number\" 19 \"Net Name\" 20} -n J1
```

## Related Commands

[ExportPinAssignmentsForConnector](#)

## ImportPlacementXMLFile

Imports the placement information for the instances from an XML format.

### Return

bool

### Syntax

```
ImportPlacementXMLFile placement_xml_file
```

## Parameters

| Parameter          | Description                                                                        | Type   | Optional |
|--------------------|------------------------------------------------------------------------------------|--------|----------|
| placement_xml_file | Specifies the path and name of the XML file that stores the placement information. | string | false    |

## Examples

- `ImportPlacementXMLFile \"D:/fsp_working/placement.xml\"`
- `GenerateDEHDLschematics`
- `GenerateLayoutData`
- `UpdateLayoutData`
- `GenerateOrCADschematics`

## Related Commands

No Related Commands

# ImportPowerMappingData

Imports the power mapping information from the external file into the design.

## Return

bool

## Syntax

```
ImportPowerMappingData powermapping_file_name
```

## Parameters

| Parameter              | Description                                                               | Type   | Optional |
|------------------------|---------------------------------------------------------------------------|--------|----------|
| powermapping_file_name | Specifies the path and name of the file that is to be used for importing. | string | no       |

## Examples

No Examples

## Related Commands

No Related Commands

# InitDesignNetNameDatabase

Updates the net names for the connected pins present in the design.

## Return

bool

## Syntax

`InitDesignNetNameDatabase`

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`InitDesignNetNameDatabase`

## Related Commands

[UpdateDeviceDataBase](#)

# InitInstancePinLocation

Updates the locations of the instance pin based on the updated placement details. Use this command if you need any canvas pin location of components and the modified components placement.

## Return

bool

## Syntax

`InitInstancePinLocation`

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`InitInstancePinLocation`

## Related Commands

- [UpdateInstanceLocation](#)
- [RotateInstance](#)
- [FlipInstance](#)
- [MoveInstance](#)

# IntListTest

Tests the integer list variable type.

## Return

int\_list

## Syntax

```
IntListTest arg
```

## Parameters

| Parameter | Description                                                    | Type     | Optional |
|-----------|----------------------------------------------------------------|----------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | int_list | false    |

## Examples

- `IntListTest [list 89 67 48]`
- `IntListTest {10 90 80}`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

# IntTest

Tests the integer variable type.

## Return

int

## Syntax

```
IntTest arg
```

## Parameters

| Parameter | Description                                                    | Type | Optional |
|-----------|----------------------------------------------------------------|------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | int  | false    |

## Examples

- `IntTest 01`
- `IntTest 999`

## Related Commands

- [BoolTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

## IsCheckSymbolLargerThanPageBorder

Returns the state of the flag for symbol size checking with the schematics page border.

### Return

bool

### Syntax

```
IsCheckSymbolLargerThanPageBorder
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
IsCheckSymbolLargerThanPageBorder
```

### Related Commands

[SetIsCheckSymbolLargerThanPageBorder](#)

## IsDraExist

Returns 1 if the specified dra exists in the paths specified in psmppath variable or else returns 0.

### Return

bool

### Syntax

```
IsDraExist draName
```

## Parameters

| Parameter | Description                                                                                      | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------|--------|----------|
| draName   | Specifies the name of the dra (with or without .dra extension) whose existence is to be checked. | string | false    |

## Examples

- IsDraExist HSTL\_X32
- IsDraExist CY7C1315BV18

## Related Commands

- [GetDraPath](#)
- [GetDraDirectoriesPaths](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## isECOMode

Returns 1 if ECO mode is on and 0 if off. ECO mode restricts pin swap functionality while performing optimization in Allegro using FSP engine and database.

## Return

bool

## Syntax

```
isECOMode
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
isECOMode
```

## Related Commands

[isECOMode](#)

## IsFPGAPinMappingFileExist

Returns one if the specified fpga pin mapping file exists in the paths specified in lrfpath variable or else returns zero.

## Return

bool

## Syntax

```
IsFPGAPinMappingFileExist draName
```

## Parameters

| Parameter                  | Description                                                                                                        | Type   | Optional |
|----------------------------|--------------------------------------------------------------------------------------------------------------------|--------|----------|
| fpga_pin_mapping_file_name | Specifies the name of the fpga pin mapping file (with or without .fpm extension) whose existence is to be checked. | string | false    |

## Examples

```
IsFPGAPinMappingFileExist ep4cgx150df27
```

## Related Commands

- [GetFPGAPinMappingPath](#)
- [GetFPGAPinMappingDirectoriesPaths](#)
- [ReportAllFPGAPinMappingFiles](#)
- [CheckDesignConsistency](#)
- [UpdateInstanceFootprint](#)
- [GetInstanceFootprint](#)

## IsGenerateSFReport

Returns the status of Synthesis Failure Report flag. Returns 1, if flag is on and 0 for off.

## Return

bool

## Syntax

```
IsGenerateSFReport
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

- `IsGenerateSFReport`

## Related Commands

[SetIsGenerateSFReport](#)

## IsNetGroupConnected

Returns 1 if at least one pin with the specified NetGroup name is connected or else returns 0.

### Return

bool

### Syntax

```
IsNetGroupConnected net_group_name
```

### Parameters

| Parameter      | Description                                                                  | Type   | Optional |
|----------------|------------------------------------------------------------------------------|--------|----------|
| net_group_name | Specifies the name of the NetGroup whose connection status is to be checked. | string | false    |

### Examples

```
IsNetGroupConnected NG1
```

### Related Commands

- [GetConnectedNetGroupNames](#)
- [GetNetGroupSize](#)

## IsOptimizeTDConnectorUtilization

Returns whether the TD connector utilization flag is on or off. When this flag is set, the FSP synthesis engine optimally spreads out the assigned pins across all the available TC connectors. If this flag is turned off, the FSP synthesis engine tries to accommodate connections in the minimum possible number of connectors.

### Return

bool

### Syntax

```
IsOptimizeTDConnectorUtilization
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
IsOptimizeTDConnectorUtilization
```



## Related Commands

[ToggleOptimizeTDConnectorUtilization](#)

## IsPerformSecondPassOptimization

Returns the status of the second pass optimization flag. Returns 1 if the flag is turned on or else returns 0 if it is off. The second pass optimization flag decides whether the synthesis would succeed with a second pass run to minimize the crossovers on the design nets.

### Return

bool

### Syntax

```
IsPerformSecondPassOptimization
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
IsPerformSecondPassOptimization
```

## Related Commands

- [SetIsPerformSecondPassOptimization](#)
- [ToggleSecondPassOptimization](#)

## IsPinNameASNetName

Returns the state of the flag for using net name as pin name during FPGA symbol generation.

### Return

bool

### Syntax

```
IsPinNameASNetName instance_name
```

### Parameters

| Parameter     | Description                                                          | Type   | Optional |
|---------------|----------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the device instance to which this flag is set. | string | no       |

### Examples

```
IsPinNameASNetName U1
```

## Related Commands

[SetPinNameASNetName](#)

# IsSkipConnectedPinsInContiguousConnections

Returns the status of a interface group, of which the connected pins should be skipped or not in contiguous connections.

## Return

bool

## Syntax

```
IsSkipConnectedPinsInContiguousConnections interface_name group_name
```

## Parameters

| Parameter      | Description                                   | Type   | Optional |
|----------------|-----------------------------------------------|--------|----------|
| interface_name | Specifies the name of the interface instance. | string | false    |
| group_name     | Specifies the name of the group.              | string | false    |

## Examples

```
IsSkipConnectedPinsInContiguousConnections U1 data1
```

## Related Commands

- [IsSkipConnectedPinsInContiguousConnectionsForProtocol](#)
- [SetSkipConnectedPins](#)

# IsSkipConnectedPinsInContiguousConnectionsForProtocol

Returns the status of a protocol group, of which the connected pins should be skipped or not in contiguous connections.

## Return

bool

## Syntax

```
IsSkipConnectedPinsInContiguousConnectionsForProtocol interface_name device_name group_name
```

## Parameters

| Parameter     | Description                                                      | Type   | Optional |
|---------------|------------------------------------------------------------------|--------|----------|
| protocol_name | Specifies the name of the protocol.                              | string | false    |
| device_name   | Specifies the name of the device to which the group connects to. | string | false    |
| group_name    | Specifies the name of the group.                                 | string | false    |

## Examples

```
IsSkipConnectedPinsInContiguousConnectionsForProtocol U1_U2_U3 U2 group2
```

## Related Commands

- [IsSkipConnectedPinsInContiguousConnections](#)
- [SetSkipConnectedPins](#)

## IsUsePart

Returns 1 if the specified instance is mapped to the schematics symbol and 0 if not mapped.

### Return

bool

### Syntax

```
IsUsePart instance_name
```

### Parameters

| Parameter     | Description                                                               | Type   | Optional |
|---------------|---------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose UsePart status is to be checked. | string | false    |

## Examples

```
IsUsePart U1
```

## Related Commands

- [GetFESymbolMapping](#)
- [GetSymbolLibraryName](#)
- [GetSymbolPartName](#)
- [GetGenerateSymbolLibraryName](#)
- [GenerateAllegroSymbol](#)
- [GenerateOrCADSymbol](#)

## LinkToFESymbol

Links the specified schematic symbol with the specified device instance. The command uses the same schematic symbol while generating schematics.

### Return

bool

### Syntax

```
LinkToFESymbol -i instance_name -s schematic_symbol_info
```

## Parameters

| Parameter | Description                                                                               | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the device instance that is to be linked with the schematic symbol. | string | false    |
| -s        | Specifies the schematic symbol information (string returned by Component Browser).        | string | false    |

## Examples

```
LinkToFESymbol -i U3 -s \"add <fsp_fe_lib>s5_4vfx100ff1152 add :%DONTANOTATE:JEDEC_TYPE=FF1152 :%DONTANOTATE:PART_NAME=S5_4VFX100FF1152
<fsp_fe_lib>s5_4vfx100ff1152.sym_1\"
```

## Related Commands

- [LinkToFESymbolOrCAD](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)

## LinkToFESymbolOrCAD

Links the specified schematic symbol with the specified device instance. The command uses the same schematic symbol while generating schematics. This command is applicable in OrCAD schematic environment.

## Return

bool

## Syntax

```
LinkToFESymbolOrCAD -i instance_name -o complete_olb_file_path -p package_name -j jedec_type
```

## Parameters

| Parameter | Description                                                                               | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the device instance that is to be linked with the schematic symbol. | string | false    |
| -o        | Complete file path of OrCAD schematic symbol library.                                     | string | false    |
| -p        | Specifies the name of the package for the specified schematic symbol.                     | string | false    |
| -j        | Specifies the name of the Jedec type for the specified schematic symbol.                  | string | false    |

## Examples

```
LinkToFESymbolOrCAD -i U3 -o \"F:/designs_data/fsp_dessigns/orcad_extern/output/OrCAD/FSP_FE_LIB.OLB\" -p 4vfx100ff1152 -j ff1152
```

## Related Commands

- [LinkToFESymbol](#)
- [ConvertLRFTToRealPart](#)
- [ConvertLRFTToRealPartOrCAD](#)
- [ConvertVIToRealInterface](#)
- [ConvertVIToRealInterfaceOrCAD](#)
- [ChangeGlobalOrCADLibraryPath](#)

## LoadProcessOptions

Imports the process settings information from the specified tag or file for design synthesis.

### Return

bool

### Syntax

```
LoadProcessOptions process_option_or_complete_file_name
```

### Parameters

| Parameter                | Description                                                                                 | Type   | Optional |
|--------------------------|---------------------------------------------------------------------------------------------|--------|----------|
| process_option_file_name | Specifies the process option tag name or path to the file that is to be used for importing. | string | false    |

### Examples

- `LoadProcessOptions run_memories_settings`
- `LoadProcessOptions /export/home/user/design_data/run_protocol.xml`
- `LoadProcessOptions ./run_memory.xml`

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [SaveProcessOptions](#)

## LoadWorkFlow

Loads the work flow from the specified xml path.

### Return

bool

## Syntax

```
LoadWorkFlow workflow_xml_filepath
```

## Parameters

| Parameter             | Description                                   | Type   | Optional |
|-----------------------|-----------------------------------------------|--------|----------|
| workflow_xml_filepath | Specifies the workflow xml path to be loaded. | string | false    |

## Examples

```
LoadWorkFlow D:/default_workflow.xml
```

## Related Commands

[SaveWorkFlowAs](#)

# LockInstanceNets

Locks all the nets that are connected to the specified instance.

## Return

bool

## Syntax

```
LockInstanceNets instance_name
```

## Parameters

| Parameter     | Description                                                                     | Type   | Optional |
|---------------|---------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance to which the connected nets is to be locked. | string | no       |

## Examples

```
LockInstanceNets U1
```

## Related Commands

[UnLockInstanceNets](#)

# LockNets

Lock nets in design. Specify at least one command option.

## Return

bool

## Syntax

```
LockNets [-all] [-clocks] [-constraint_pins] [-nets {list_of_net_names}] [-bus bus_name] [-inst_or_protocol instance_or_protocol_name] [-inst_or_protocol_group_or_bank group_or_bank_name]
```

## Parameters

| Parameter                       | Description                                                                                   | Type        | Optional |
|---------------------------------|-----------------------------------------------------------------------------------------------|-------------|----------|
| -all                            | Locks all the nets.                                                                           | bool        | true     |
| -clocks                         | Locks all the clock nets.                                                                     | bool        | true     |
| -constraint_pins                | Locks all the constraint pin nets.                                                            | bool        | true     |
| -nets                           | Locks all the specified nets.                                                                 | string_list | true     |
| -bus                            | Locks all the nets belonging to specified bus name.                                           | string      | true     |
| -inst_or_protocol               | Locks all the nets belonging to specified interface or protocol.                              | string      | true     |
| -inst_or_protocol_group_or_bank | Locks all the nets belonging to specified group (or bank) of specified interface or protocol. | string      | true     |

## Examples

- `LockNets -all`
- `LockNets -clocks`
- `LockNets -clocks -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `LockNets -clocks -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `LockNets -clocks -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `LockNets -constraint_pins`
- `LockNets -constraint_pins -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `LockNets -constraint_pins -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `LockNets -constraint_pins -inst_or_protocol U1 U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `LockNets -nets {XP1_DDR_A0,XP1_DDR_A1}`
- `LockNets -bus XP1_DDR_A`
- `LockNets -inst_or_protocol U1`
- `LockNets -inst_or_protocol U1_U3`
- `LockNets -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `LockNets -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `LockNets -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`

## Related Commands

- [UnlockNets](#)
- [GetAllLockedNetNames](#)

## MapConnectorPinAssignment

This is an internal command. Updates the connector pin assignment by importing the CSV file.

## Return

bool

## Syntax

```
MapConnectorPinAssignment -c csv_file_path -d delimiter -m column_mapping -r reference_column_name -a connector_name [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                    | Type              | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -c        | Specifies the file path that is to used to import the connector pin assignment.                                                                | string            | no       |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), '(space) etc..                 | string            | no       |
| -m        | Specifies the map between the column name to column number.                                                                                    | string_string_map | no       |
| -r        | Specifies the name of the reference column.                                                                                                    | string            | no       |
| -a        | Specifies the name of the connector that is be updated.                                                                                        | string            | no       |
| -i        | Specifies a list of row numbers, in comma separated format, to be ignored from the CSV file. By default, the command does not ignore any rows. | string            | yes      |

## Examples

```
MapConnectorPinAssignment -c ./update_connector_csv.csv -d , -i 1 -r "Signal Name" -a J1 -m {"Signal Name" 1 "Interface/Protocol Name" 2
"Assigned to Pin" 4}
```

## Related Commands

[ExportCSVFromConnectorMapping](#)

# MapPortNamestoPinNames

Assigns the port names to pin names.

## Return

bool

## Syntax

```
MapPortNamestoPinNames -i interface_name -d device_name -pp pin_name_vs_port_name_map
```

## Parameters

| Parameter | Description                                                              | Type              | Optional |
|-----------|--------------------------------------------------------------------------|-------------------|----------|
| -i        | Specifies the name of the interface for which the mapping is to be done. | string            | false    |
| -d        | Specifies the name of the device.                                        | string            | false    |
| -pp       | Specifies the pin name to port name map.                                 | string_string_map | false    |



## Examples

```
MapPortNameToPinNames -i XP1 -d U1 -pp {DDR_DM0 XP1_U1_U2_DDR_DQs0 DDR_DQS0 XP1_U1_U2_DDR_DQS0}
```

## Related Commands

# MapPortNameToNetName

Maps the port names to the net names.

## Return

bool

## Syntax

```
MapPortNameToNetName -i interface_name -d device_name -np net_name_vs_port_name_map
```

## Parameters

| Parameter | Description                                                              | Type              | Optional |
|-----------|--------------------------------------------------------------------------|-------------------|----------|
| -i        | Specifies the name of the interface for which the mapping is to be done. | string            | false    |
| -d        | Specifies the name of the device instance.                               | string            | false    |
| -np       | Specifies the net name to port name map.                                 | string_string_map | false    |

## Examples

```
MapPortNameToNetName -i XP1 -d U1 -np {XP1_U1_U2_DDR_DM0 XP1_U1_U2_DDR_DM0 XP1_U1_U2_DDR_DQs0 XP1_U1_U2_DDR_DQS0}
```

## Related Commands

- [MapPortNameToPinNames](#)

# MapPowerFilterToInstancePin

Assigns the specified power filter to the pin of the specified instance.

## Return

bool

## Syntax

```
MapPowerFilterToInstancePin instance_name pin_number termination_name power_reg_name
```

## Parameters

| Parameter        | Description                                                                           | Type   | Optional |
|------------------|---------------------------------------------------------------------------------------|--------|----------|
| instance_name    | Specifies the name of the instance.                                                   | string | false    |
| pin_number       | Specifies the pin number on which the specified power filter is to be assigned.       | string | false    |
| termination_name | Specifies the name of the power filter that is to be assigned on the instance pin.    | string | false    |
| power_reg_name   | Specifies the name of the power regulator that is to be assigned on the power filter. | string | false    |

## Examples

- `MapPowerFilterToInstancePin U5 J3 power_filtr GND`
- `MapPowerFilterToInstancePin XP1 100 power_filtr V_2_5`

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapTerminationToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [SpecifyDiffPinTermination](#)

## MapResources

Attaches the resource information specified to device instance or interface instance or protocol.

## Return

bool

## Syntax

```
MapResources -r resource -d deviceInstanceName [-i interfaceInstanceName]
```

## Parameters

| Parameter | Description                                                                                                                                                             | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -r        | Specifies the information of the resource that is to be attached.                                                                                                       | string | false    |
| -d        | Specifies the name of the device instance.                                                                                                                              | string | false    |
| -i        | Specifies the name of the interface instance or protocol. In case argument is not specified, the command maps the specified resources to the specified device instance. | string | true     |

## Examples

```
MapResources -r \"CONFIG_PROHIBIT = A31\" -d U1
```

## Related Commands

- [GetResources](#)

## MapTermination

Maps the termination to the specified schematic symbol.

### Return

bool

### Syntax

```
MapTermination -s schematic_symbol_info -m port_mapping -n termination_name [-high high_regulator_name] [-low low_regulator_name] [-p termination_part_number]
```

### Parameters

| Parameter | Description                                                                                                                                                                       | Type              | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -n        | Specifies the name of the termination that is to be mapped with the schematic symbol.                                                                                             | string            | false    |
| -s        | Specifies the information of the schematic symbol. This string is returned from the Component Browser.                                                                            | string            | false    |
| -m        | Specifies the termination port mapping. Examples, P1,a1; P2,a2; Here P1, P2 are FSP template port names while a1, a2 are port names of mapped termination.                        | string_string_map | false    |
| -high     | Specifies the name of the Supply or High regulator for the Pull Up/Down termination.                                                                                              | string            | true     |
| -low      | Specifies the name of the Ground or Low regulator for the Pull Up/Down termination.                                                                                               | string            | true     |
| -p        | The number (or position) of the termination component for which mapping is specified. Use 1 for first discrete and 2 for second discrete used in termination. Default value is 1. | int               | true     |

### Examples

```
MapTermination -n ser_term -s "\"add <classlib>res add :%Value:FSP_KEY_1=4 :%Value:FSP_KEY_2=4 :%Value:PACK_TYPE=SMDRES :%Value:VALUE=100 :%Value:TOLERANCE=2% :%DONTANOTATE:JEDEC_TYPE=SM_0805 :%DONTANOTATE:DESCRIPTION=resistor 100 2% :%DONTANOTATE:VALUE=100 :%DONTANOTATE:PART_NUMBER=res456 :%DONTANOTATE:TOLERANCE=2% :%DONTANOTATE:PART_NAME=RES <classlib>res.sym_1\" -m \"P1 A P2 B\""
```

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTerminationOrCAD](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)

## MapTerminationOrCAD

Maps the termination to the specified schematic symbol.

## Return

bool

## Syntax

```
MapTerminationOrCAD -s schematic_symbol_info -m port_mapping -n termination_name [-high high_regulator_name]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -n        | Specifies the name of the termination that is to be mapped to the specified schematic symbol.                                                                  | string            | false    |
| -o        | Specifies the name of the OrCAD symbol library file.                                                                                                           | string            | false    |
| -p        | Specifies the name of the OrCAD package.                                                                                                                       | string            | false    |
| -m        | Specifies the termination the port mapping. Examples, P1,a1; P2,a2; Here P1, P2 are FSP template port names while a1, a2 are port names of mapped termination. | string_string_map | false    |
| -high     | Specifies the name of the Supply or High regulator for the Pull Up/Down termination.                                                                           | string            | true     |

## Examples

```
MapTerminationOrCAD -n ser_um -o %cdsroot%/tools/capture/library/Discrete.olb -p R -m \"P1 2 P2 1\"
```

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [GetTerminationNames](#)
- [SpecifyDiffPinTermination](#)
- [ChangeGlobalOrCADLibraryPath](#)

## MapTerminationToInstancePin

Assigns the specified termination to the pins of the specified interface instance.

## Return

bool

## Syntax

```
MapTerminationToInstancePin instance_name pin_number termination_name
```

## Parameters

| Parameter        | Description                                                                                    | Type   | Optional |
|------------------|------------------------------------------------------------------------------------------------|--------|----------|
| instance_name    | Specifies the name of instance for which the specified termination is to be assigned.          | string | false    |
| pin_number       | Specifies the pin number of the instance on which the specified termination is to be assigned. | string | false    |
| termination_name | Specifies the name of the termination that is to be assigned on the instance pin.              | string | false    |

## Examples

- `MapTerminationToInstancePin XP1 153 ser_term`
- `MapTerminationToInstancePin XP1 177 thev_term`

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [SpecifyDiffPinTermination](#)

# MapTerminationToInstancePinOtherEnd

Assigns the specified termination to the other end of the specified interface instance pin. The termination will be connected to targeted device pin when net is synthesized in design.

## Return

bool

## Syntax

```
MapTerminationToInstancePinOtherEnd instance_name pin_number termination_name
```

## Parameters

| Parameter        | Description                                                                               | Type   | Optional |
|------------------|-------------------------------------------------------------------------------------------|--------|----------|
| instance_name    | Specifies the name of the instance for which the specified termination is to be assigned. | string | false    |
| pin_number       | Specifies the pin number on which the specified termination is to be assigned.            | string | false    |
| termination_name | Specifies the name of the termination that is to be assigned to the instance pin.         | string | false    |

## Examples

- `MapTerminationToInstancePinOtherEnd XP1 153 ser_term`
- `MapTerminationToInstancePinOtherEnd XP1 177 thev_terms`

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapTerminationToInstancePin](#)
- [MapPowerFilterToInstancePin](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)
- [SpecifyDiffPinTermination](#)

## MergeAllSymbolSplits

Creates a single symbol split by merging all the symbol splits.

### Return

bool

### Syntax

```
MergeAllSymbolSplits instance_name
```

### Parameters

| Parameter     | Description                                                               | Type   | Optional |
|---------------|---------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance for which the symbols is to be merged. | string | false    |

### Examples

```
MergeAllSymbolSplits U2
```

## Related Commands

- [SplitSymbolConnectionWise](#)
- [SplitSymbolBankWise](#)
- [AutoSplitSymbol](#)

## MoveDeviceNet

Moves the net from the current specified pin to the new specified pin within an instance. This command does not allow you to move a net from one instance to another instance.

### Return

bool

### Syntax

```
MoveDeviceNet instance_name fromPinNumber toPinNumber
```

## Parameters

| Parameter     | Description                                                         | Type   | Optional |
|---------------|---------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance on which the net is to be moved. | string | false    |
| fromPinNumber | Specifies the pin number on which the net is connected.             | string | false    |
| toPinNumber   | Specifies the pin number on which the net is to be moved.           | string | false    |

## Examples

```
MoveDeviceNet U2 E3 E4
```

## Related Commands

- [MovePinNet](#)

# MoveInstance

Repositions the specified instance from the current location to the specified location.

## Return

bool

## Syntax

```
MoveInstance instance_name centerXLoc centerYLoc
```

## Parameters

| Parameter     | Description                                    | Type   | Optional |
|---------------|------------------------------------------------|--------|----------|
| instance_name | Name of the instance that need is to be moved. | string | false    |
| centerXLoc    | Specifies the center X coordinate.             | double | false    |
| centerYLoc    | Specifies the center Y coordinate.             | double | false    |

## Examples

```
MoveInstance U2 5 5
```

## Related Commands

- [UpdateInstanceLocation](#)
- [RotateInstance](#)
- [FlipInstance](#)
- [InitInstancePinLocation](#)

# MovePinNet

Moves the net from the current specified instance pin to the new specified instance pin. This command allows you to move a net from one instance to another instance.

## Return

bool

## Syntax

```
MovePinNet from_instance_name from_pin_number to_instance_name to_pin_number
```

## Parameters

| Parameter          | Description                                                                     | Type   | Optional |
|--------------------|---------------------------------------------------------------------------------|--------|----------|
| from_instance_name | Specifies the name of the source instance for which the net is to be updated.   | string | false    |
| from_pin_number    | Specifies the pin number on which the net is connected.                         | string | false    |
| to_instance_name   | Specifies the name of the destination instance on which the net is to be moved. | string | false    |
| to_pin_number      | Specifies the pin number on which the net is to be moved.                       | string | false    |

## Examples

- `MovePinNet U2 E3 U3 E4`
- `MovePinNet J1 A1 J2 A2`

## Related Commands

- [MoveDeviceNet](#)

## NewProject

Creates a FSP project at the specified path with the specified name.

## Return

bool

## Syntax

```
NewProject projectPath projectName
```

## Parameters

| Parameter   | Description                                             | Type   | Optional |
|-------------|---------------------------------------------------------|--------|----------|
| projectPath | Path to the directory where you want to create project. | string | false    |
| projectName | Specifies the name of the project.                      | string | false    |

## Examples

```
NewProject C:/SPB_Data/fsp_working new_pro_test
```



## Related Commands

[CreateNewProject](#)

## OpenDEHDLProject

Opens the DE HDL project using the .cpm file. If the associated fsp project file is not present in the fsp folder a blank new project is created under the root design mentioned in the .cpm file.

## Return

bool

## Syntax

```
OpenDEHDLProject [-cpm cpm_file_Name]
```

## Parameters

| Parameter | Description                                   | Type   | Optional |
|-----------|-----------------------------------------------|--------|----------|
| -cpm      | Specifies the path and name of the .cpm file. | string | false    |

## Examples

```
OpenDEHDLProject -cpm C:/SPB_Data/fsp_working/tcl_test_setup/tcl_command.cpm
```

## Related Commands

- [CreateNewDEHDLProject](#)
- [OpenProject](#)

## OpenProject

Opens the specified project in FSP.

## Return

bool

## Syntax

```
OpenProject projectFilePathName
```

## Parameters

| Parameter           | Description                                                      | Type   | Optional |
|---------------------|------------------------------------------------------------------|--------|----------|
| projectFilePathName | Specifies the path to the FSP project file that is to be opened. | string | false    |

## Examples

```
OpenProject C:/SPB_Data/fsp_working/project4/project4.fsp
```

## Related Commands

- [GetProjectFilePath](#)
- [GetProjectName](#)
- [CloseProject](#)
- [SaveProject](#)
- [SaveProjectAs](#)

## PlaceInstance

Places the specified rules part in the Canvas.

### Return

string

### Syntax

```
PlaceInstance -lrf rules_file [-xloc x_location] [-yloc y_location]
```

### Parameters

| Parameter | Description                                                                                                                         | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -rules    | Specifies the name of the rules file that is to be placed in the Canvas.                                                            | string | false    |
| -xloc     | Specifies the bottom-left X location where the part is to be placed. If not specified, part will be placed with bottom left x at 0. | double | true     |
| -yloc     | Specifies the bottom-left Y location where the part is to be placed. If not specified, part will be placed with bottom left y at 0. | double | true     |

### Examples

- `PlaceInstance -rules 4vfx40ffl152 -xloc 5 -yloc 5`
- `PlaceInstance -rules 4vfx40ffl152`

## Related Commands

- [AddPart](#)
- [AddPartOrCAD](#)
- [AddPartModel](#)

## PreservePairPins

Marks all the differential pair pins of the specified device instance as preserve pins.

### Return

bool

## Syntax

```
PreservePairPins device_name
```

## Parameters

| Parameter          | Description                                                                                                 | Type   | Optional |
|--------------------|-------------------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance of which differential pair pins is to be marked as preserve pins. | string | false    |

## Examples

```
PreservePairPins U1
```

## Related Commands

- [SetPreserveUnusedPinsInBank](#)
- [ResetPreserveUnusedPinsInBank](#)
- [PreserveTrueDifferentialPins](#)
- [SetPreservePins](#)
- [SetDevicePreservePins](#)
- [GetPreservePins](#)

## PreserveTrueDifferentialPins

Preserves the true differential pins of the specified device instance. This command is applicable to Actel devices.

## Return

bool

## Syntax

```
PreserveTrueDifferentialPins device_instance_name
```

## Parameters

| Parameter          | Description                                | Type   | Optional |
|--------------------|--------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance. | string | false    |

## Examples

```
PreserveTrueDifferentialPins U3
```

## Related Commands

[UnPreserveTrueDifferentialPins](#)

## Quit

Closes the FSP.

## Return

void

## Syntax

```
Quit
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
Quit
```

## Related Commands

- [OpenProject](#)
- [CloseProject](#)
- [SaveProject](#)

# RemoveAllProcessOptions

Removes all process options that are defined in the Process Option Editor.

## Return

bool

## Syntax

```
RemoveAllProcessOptions
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
RemoveAllProcessOptions
```

## Related Commands

- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [LoadProcessOptions](#)
- [SaveProcessOptions](#)

## RemoveAllProtocols

Removes all the device protocols from the current design.

### Return

bool

### Syntax

```
RemoveAllProtocols
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
RemoveAllProtocols
```

### Related Commands

- [RemoveProtocol](#)
- [GetProtocolNames](#)
- [RenameProtocol](#)
- [DeleteInstance](#)
- [DeleteNets](#)

## RemoveDeepNWideGroup

Removes the specified Deep and Wide group.

### Return

bool

### Syntax

```
RemoveDeepNWideGroup dnw_group_name
```

### Parameters

| Parameter      | Description                                                          | Type   | Optional |
|----------------|----------------------------------------------------------------------|--------|----------|
| dnw_group_name | Specifies the name of the Deep and Wide group that is to be removed. | string | false    |

### Examples

- `RemoveDeepNWideGroup common_group1`
- `RemoveDeepNWideGroup wide_bus1`

## Related Commands

- [CreateCommonGroup](#)
- [GetDeepNWideGroups](#)
- [UpdateDeepNWideGroupName](#)
- [GetDeepNWideGroupInstanceList](#)

## RemoveProcessOption

Remove existing process option.

## Return

bool

## Syntax

```
RemoveProcessOption processOptionTagName
```

## Parameters

| Parameter            | Description                                                     | Type   | Optional |
|----------------------|-----------------------------------------------------------------|--------|----------|
| processOptionTagName | Specifies the name of the process option that is to be removed. | string | false    |

## Examples

- `RemoveProcessOption run_protocol`
- `RemoveProcessOption run_memory`

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [LoadProcessOptions](#)
- [RenameProcessOption](#)
- [RemoveAllProcessOptions](#)

## RemoveProtocol

Removes the specified protocol from the current design.

## Return

bool

## Syntax

```
RemoveAllProtocol protocol_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| protocol_name | Specifies the name of the protocol that is to be removed. | string | false    |

## Examples

```
RemoveProtocol U3,U4
```

## Related Commands

- [GetProtocolNames](#)
- [RenameProtocol](#)
- [RemoveAllProtocols](#)
- [DeleteInstance](#)
- [DeleteNets](#)

## RemoveRulesFilePath

Deletes the specified rules directory path from the rules search path.

## Return

bool

## Syntax

```
RemoveRulesFilePath rules_dir_path
```

## Parameters

| Parameter      | Description                                           | Type   | Optional |
|----------------|-------------------------------------------------------|--------|----------|
| rules_dir_path | Specifies the rules files path that is to be removed. | string | false    |

## Examples

```
RemoveRulesFilePath \"D:/fsp_working/my_rules\"
```

## Related Commands

- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [GetRulesWorkingDir](#)
- [SetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [CheckDesignConsistency](#)

## RemoveSearchNReplaceNetNamePatterns

Removes the patterns defined for the net names.

### Return

bool

### Syntax

```
RemoveSearchNReplaceNetNamePatterns
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

No Examples

## Related Commands

- [GetSearchNReplaceNetNamePattern](#)
- [AddSearchNReplaceNetNamePattern](#)
- [SetSearchNReplaceNetNamePattern](#)

## RenameInstance

Changes the name of the specified instance.

### Return

bool

### Syntax

```
RenameInstance presentInstanceName newInstanceName
```



## Parameters

| Parameter           | Description                                                   | Type   | Optional |
|---------------------|---------------------------------------------------------------|--------|----------|
| presentInstanceName | Specifies the existing instance name.                         | string | false    |
| newInstanceName     | Specifies that name to be assigned to the specified instance. | string | false    |

## Examples

- `RenameInstance U1 MY_U1`
- `RenameInstance XP56 I10`

## Related Commands

[DeleteInstance](#)

# RenameNetGroup

Changes the name of the specified NetGroup.

## Return

bool

## Syntax

```
RenameNetGroup -o old_net_group_name -n new_net_group_name
```

## Parameters

| Parameter | Description                                                                 | Type   | Optional |
|-----------|-----------------------------------------------------------------------------|--------|----------|
| -o        | Specifies the name of the existing NetGroup.                                | string | false    |
| -n        | Specifies the name of the NetGroup to be assigned to the existing NetGroup. | string | false    |

## Examples

- `RenameNetGroup NetGroup1 NG1`
- `RenameNetGroup NetGroup[1:6] NG1`
- `RenameNetGroup NetGroup* NG1`

## Related Commands

- [DeleteNetGroups](#)
- [CreateNewNetGroup](#)

# RenamePowerRegulator

Changes the name of the specified power regulator.

## Return

bool

## Syntax

```
RenamePowerRegulator present_regulator_name new_regulator_name
```

## Parameters

| Parameter              | Description                           | Type   | Optional |
|------------------------|---------------------------------------|--------|----------|
| present_regulator_name | Name of the existing power regulator. | string | false    |
| new_regulator_name     | New name for the power regulator.     | string | false    |

## Examples

```
RenamePowerRegulator V_0_9 V_0_9_EXT
```

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

# RenameProcessOption

Renames the existing name of the process option to a new name.

## Return

bool

## Syntax

```
RenameProcessOption oldProcessOptionTagName newProcessOptionTagName
```

## Parameters

| Parameter               | Description                                              | Type   | Optional |
|-------------------------|----------------------------------------------------------|--------|----------|
| oldProcessOptionTagName | Specifies the existing name of the process option name.  | string | false    |
| newProcessOptionTagName | Specifies the name to be assigned to the process option. | string | false    |

## Examples

- `RenameProcessOption run_protocol run_all_protocol`
- `RenameProcessOption run_memory run_ddr3_memory`

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [LoadProcessOptions](#)
- [RemoveProcessOption](#)
- [RemoveAllProcessOptions](#)

## ReOptimizeProtocol

Re-optimizes the specified protocol connections for the specified device instance.

### Return

int

### Syntax

```
ReOptimizeProtocol target_device_instance_name reference_device_instance_name protocol_name
```

### Parameters

| Parameter                      | Description                                                                                                                                                                                                                                                                                                                                                                           | Type   | Optional |
|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| target_device_instance_name    | Specifies the name of the device instance for which the protocol connections is to be optimized.                                                                                                                                                                                                                                                                                      | string | false    |
| reference_device_instance_name | Specify the reference device instance name from which protocol connectivity needs to be optimized. This field is useful when protocol is connected to more than 2 devices. For example, for protocol U1,U2,U3,U4,U5, connectivity for U3 can be optimized from reference device U2 or U4. Considering the chain in given order, U2 is the ideal choice for optimization in such case. | string | false    |
| protocol_name                  | Specifies name of the protocol that is to be optimized.                                                                                                                                                                                                                                                                                                                               | string | false    |

## Examples

- `ReOptimizeProtocol U3 U2 U1,U2,U3,U4,U5`
- `ReOptimizeProtocol U10 U9 U8,U9,U10,U11`

## Related Commands

- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ExportCSVFromProtocol](#)
- [UpdateProtocolFromCSV](#)
- [ExportProtocolDefinition](#)
- [SetAutoNetGroupProtocol](#)

## ReplaceFPGA

Replaces the existing device instance with the specified new device instance. Command allows to switch to different FPGA model only within same family.

### Return

bool

### Syntax

```
ReplaceFPGA deviceInstanceName newFPGAPartName
```

### Parameters

| Parameter          | Description                                                            | Type   | Optional |
|--------------------|------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance that is to be replaced.      | string | false    |
| newFPGAPartName    | Specifies the name of the FPGA rules file that is to be replaced with. | string | false    |

### Examples

```
ReplaceFPGA U5 4vfx12ff668
```

## Related Commands

- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)

## ReportAllDRAFiles

Reports all dra directory and file paths (fetch from psmopath) accessible to FSP.

### Return

bool

### Syntax

```
ReportAllDRAFiles
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ReportAllDRAFiles
```

## Related Commands

- [ReportAllMappingFiles](#)
- [ReportAllRulesFiles](#)
- [ReportDesignFileReferences](#)
- [CheckDesignConsistency](#)
- [IsDraExist](#)
- [GetDraPath](#)
- [GetDraDirectoriesPaths](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [CheckDesignConsistency](#)

# ReportAllFPGAPinMappingFiles

Reports all fpga pin mapping directory and file paths (fetch from \$Irfpath) that are accessible by FSP.

## Return

bool

## Syntax

```
ReportAllFPGAPinMappingFiles
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ReportAllFPGAPinMappingFiles
```

## Related Commands

- [IsFPGAPinMappingFileExist](#)
- [GetFPGAPinMappingPath](#)
- [ReportAllFPGAPinMappingFiles](#)
- [CheckDesignConsistency](#)

## ReportAllMappingFiles

Reports all mapping directory and file paths (fetch from design rules path) accessible to FSP.

### Return

bool

### Syntax

```
ReportAllMappingFiles
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
ReportAllMappingFiles
```

### Related Commands

- [ReportAllDRAFiles](#)
- [ReportAllRulesFiles](#)
- [ReportDesignFileReferences](#)
- [CheckDesignConsistency](#)

## ReportAllRulesFiles

Reports all rules directory and file paths (fetch from design rules path) accessible to FSP.

### Return

bool

### Syntax

```
ReportAllRulesFiles
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
ReportAllRulesFiles
```

## Related Commands

- [ReportAllDRAFiles](#)
- [ReportAllMappingFiles](#)
- [ReportDesignFileReferences](#)
- [CheckDesignConsistency](#)
- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [RemoveRulesFilePath](#)
- [GetRulesWorkingDir](#)
- [SetRulesWorkingDir](#)
- [GetResolvedRulesFilePaths](#)

## ReportDesign

Displays the design connectivity information and instance property information in the Log window.

### Return

bool

### Syntax

```
ReportDesign
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
ReportDesign
```

## Related Commands

- [ReportDesignToFile](#)
- [ReportInstance](#)
- [ReportInstanceToFile](#)

## ReportDesignFileReferences

Reports the dra, rules, mapping, and schematic symbol files that are being used in design.

### Return

bool

### Syntax

```
ReportDesignFileReferences
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`ReportDesignFileReferences`

## Related Commands

- [ReportAllDRAFiles](#)
- [ReportAllMappingFiles](#)
- [ReportAllRulesFiles](#)
- [CheckDesignConsistency](#)

# ReportDesignTermination

Generates a report of the terminations used in the design and related connectivity information.

## Return

void

## Syntax

`ReportDesignTermination [file_path]`

## Parameters

| Parameter              | Description                                                        | Type   | Optional |
|------------------------|--------------------------------------------------------------------|--------|----------|
| <code>file_path</code> | Specifies absolute or relative file path where report to be saved. | string | true     |

## Examples

- `ReportDesignTermination`
- `ReportDesignTermination ./output/termination_report.txt`

## Related Commands

- [ReportDesignToFile](#)
- [ReportInstance](#)
- [ReportInstanceToFile](#)

# ReportDesignToFile

Exports the design connectivity and instance property information to the `$project_dir/report.txt` file.

## Return

bool



## Syntax

`ReportDesignToFile`

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

`ReportDesignToFile`

## Related Commands

- [ReportDesign](#)
- [ReportInstance](#)
- [ReportInstanceToFile](#)

# ReportInstance

Displays the property and connectivity status of the specified instance in the Log window.

## Return

bool

## Syntax

`ReportInstance instance_name`

## Parameters

| Parameter     | Description           | Type   | Optional |
|---------------|-----------------------|--------|----------|
| instance_name | Name of the instance. | string | false    |

## Examples

- `ReportInstance U1`
- `ReportInstance XP2`

## Related Commands

- [ReportDesign](#)
- [ReportDesignToFile](#)
- [ReportInstanceToFile](#)

# ReportInstanceToFile

Exports the connectivity and property status of the specified instance to the `$project_dir/reports/$instance_name.txt` file.

## Return

bool

## Syntax

```
ReportInstanceToFile instance_name
```

## Parameters

| Parameter     | Description                                               | Type   | Optional |
|---------------|-----------------------------------------------------------|--------|----------|
| instance_name | Specifies instance name for which report to be generated. | string | false    |

## Examples

- `ReportInstanceToFile U1`
- `ReportInstanceToFile XP2`

## Related Commands

- [ReportDesign](#)
- [ReportDesignToFile](#)
- [ReportInstance](#)

## ResetAllDoNotConnect

Unmarks all the pins that are marked as do not connect of the specified instance.

## Return

bool

## Syntax

```
ResetAllDoNotConnect instance_name
```

## Parameters

| Parameter     | Description                         | Type   | Optional |
|---------------|-------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance. | string | false    |

## Examples

```
ResetAllDoNotConnect XP1
```

## Related Commands

- [GetAllDoNotConnect](#)
- [GetDoNotConnect](#)
- [ResetDoNotConnectPin](#)
- [SetDoNotConnectPin](#)

## ResetAllTerminationsRefDes

Removes the reference designators from all the terminations in the current design.

## Return

void

## Syntax

```
ResetAllTerminationsRefDes
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ResetAllTerminationsRefDes
```

## Related Commands

- [ImportFromAllegroBoard](#)
- [GenerateDEHDL Schematics](#)
- [GenerateOrCAD Schematics](#)

## ResetAssignedToPins

This command resets the Assigned to Pins on the targeted interfaces/protocols/virtual interfaces of the specified device/connector. You can optionally restrict the scope of this operation to a specific interface/protocol/virtual interface or a specific logical group under them using the optional arguments.

## Return

bool

## Syntax

```
ResetAssignedToPins [-d device_or_connector_name] [-i interface_or_protocol_or_vi_name] [-g comma_separated_group_names]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                                                                                                                           | Type   | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device or connector for which the Assigned to Pins on the targeted interfaces/protocols/virtual interfaces have to be reset. If this argument is not specified, then the reset operation is not restricted to the interfaces/protocols/virtual interfaces targeted to the specified device. | string | true     |
| -i        | Specifies the name of the interface, protocol or virtual interface on which the Assigned to Pins have to be reset. If this argument is not specified, then the reset operation is not restricted to this interface/protocol/virtual interface.                                                                        | string | true     |
| -g        | Specifies the name of the group on which the Assigned to Pins have to be reset. If this argument is not specified, then the reset operation is not restricted to this group.                                                                                                                                          | string | true     |

## Examples

- `ResetAssignedToPins`
- `ResetAssignedToPins -d U2`
- `ResetAssignedToPins -d U2 -i XP1`
- `ResetAssignedToPins -d U2 -i XP1 -g Address_control`

## Related Commands

[UpdateAssignedToPinsWithConnectedPins](#)

# ResetCachedNetNames

Resets the names that are cached in the Net column in Design Connectivity. This command is valid only for pins which are not routed.

## Return

bool

## Syntax

```
ResetCachedNetNames [interface_or_vi_or_protocol_name]
```

## Parameters

| Parameter                        | Description                                                                                                                                                                                               | Type   | Optional |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| interface_or_vi_or_protocol_name | Specifies the name of the interface instance or protocol whose net names is to be removed. If this argument is not specified, then the cached signal names for all instances and protocols will be reset. | string | true     |

## Examples

- `ResetCachedNetNames U5`
- `ResetCachedNetNames U1_U2`

## Related Commands

[SetCachedNetNames](#)

## ResetDesignPowerMapping

Removes power connections on all the power pins in the design.

### Return

bool

### Syntax

```
ResetDesignPowerMapping
```

### Parameters

No Parameters

### Examples

```
ResetDesignPowerMapping
```

### Related Commands

[ResetInstancePowerMapping](#)

## ResetDoNotConnectPin

Resets the specified pins that are do not connected. This indicates after running this command the specified pins will be available for connection.

### Return

bool

### Syntax

```
ResetDoNotConnectPin instance_name {pin_number_list}
```

### Parameters

| Parameter       | Description                                                                         | Type        | Optional |
|-----------------|-------------------------------------------------------------------------------------|-------------|----------|
| instance_name   | Specifies the name of the instance.                                                 | string      | false    |
| pin_number_list | Specifies a list of pin numbers that are in do not connect mode and is to be reset. | string_list | false    |

### Examples

```
ResetDoNotConnectPin J1 {1 2 3 4 5 6 7}
```

## Related Commands

- [GetAllDoNotConnect](#)
- [GetDoNotConnect](#)
- [SetDoNotConnectPin](#)
- [ResetAllDoNotConnect](#)

## ResetExternPin

Resets the specified pin as non external pin.

### Return

bool

### Syntax

```
ResetExternPin instance_name pin_number
```

### Parameters

| Parameter     | Description                                                                     | Type   | Optional |
|---------------|---------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                                             | string | false    |
| pin_number    | Specifies the pin number of the pin that you want to reset as non external pin. | string | false    |

### Examples

- `ResetExternPin J1 A15`
- `ResetExternPin U6 1`

## Related Commands

[SetExternPin](#)

## ResetFSPRegistry

Clears the selective or complete registry settings for FSP. Registry settings contains dialog size, dockable widget positions, TCL commands history, file paths specified by you, column visibilities, and order.

### Return

bool

### Syntax

```
ResetFSPRegistry [-report] [-all] [-key key_string]
```

## Parameters

| Parameter | Description                                     | Type   | Optional |
|-----------|-------------------------------------------------|--------|----------|
| -report   | Reports all registry key settings               | bool   | true     |
| -all      | Use this option to reset all registry settings. | bool   | true     |
| -key      | Specifies a registry key to remove.             | string | true     |

## Examples

- `ResetFSPRegistry -report`
- `ResetFSPRegistry -all`
- `ResetFSPRegistry -key UserSettings/MainWindowStates`

## Related Commands

No Related Commands

# ResetInstancePowerMapping

Removes power connections on all power pins of the specified instance.

## Return

bool

## Syntax

`ResetInstancePowerMapping`

## Parameters

| Parameter     | Description                                                            | Type   | Optional |
|---------------|------------------------------------------------------------------------|--------|----------|
| instance_name | Name of the instance for which the power connections is to be removed. | string | false    |

## Examples

- `ResetInstancePowerMapping U1`
- `ResetInstancePowerMapping XP1`

## Related Commands

[ResetDesignPowerMapping](#)

# ResetMappedPorts

Removes the name of the RTL port defined for signals of the specified interface instance.

## Return

bool

## Syntax

```
ResetMapppedPorts interfaceInstanceName
```

## Parameters

| Parameter     | Description                                                                                | Type   | Optional |
|---------------|--------------------------------------------------------------------------------------------|--------|----------|
| interfaceName | Specifies the name of the interface instance of which the RTL port names is to be removed. | string | false    |

## Examples

```
ResetMapppedPorts U2
```

## Related Commands

[AutoUpdateDeviceFPGAPortNames](#)

# ResetPDCPreservedVREFs

Removes the PDC specified VREFs from the device. This command will remove all the PDC preserved VREFs from device instance in case no arguments is specified.

## Return

bool

## Syntax

```
ResetPDCPreservedVREFs -d device_inst_name [-vref {list_of_vref_pin_numbers}]
```

## Parameters

| Parameter | Description                                                                                                                                                  | Type        | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -d        | Specifies the name of the device instance whose VREF pins is to be updated.                                                                                  | string      | false    |
| -vref     | Specifies the vref pin numbers of the PDC that is to be reset. In case no vref pin numbers are specified, this command will reset PDC for all the vref pins. | string_list | true     |

## Examples

- ResetPDCPreservedVREFs -device U1
- ResetPDCPreservedVREFs -device U2 -vref [list A21 E11]

## Related Commands

[ImportPDC](#)



## ResetPowerMapping

Removes power connections on all power pins of the specified instance or group/bank or all instances.

### Return

bool

### Syntax

```
ResetPowerMapping [-instances {instance_name_list}] [-group_or_banks {group_or_bank_name_list}]
```

### Parameters

| Parameter       | Description                                                                                                                                                          | Type        | Optional |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -instances      | list of instance names for which the power connections is to be removed. Incase not specified, power connections will be removed for all instances in design.        | string_list | true     |
| -group_or_banks | list of group (or bank) names for which the power connections is to be removed. Incase not specified, power connections will be removed for all instances in design. | string_list | true     |

### Examples

- `ResetPowerMapping`
- `ResetPowerMapping -instances U1`
- `ResetPowerMapping -instances U1 -group_or_banks bank1`
- `ResetPowerMapping -instances U1 -group_or_banks [list bank1 bank2]`
- `ResetPowerMapping -instances XP1`
- `ResetPowerMapping -instances XP1 -group_or_banks group1`
- `ResetPowerMapping -instances XP1 -group_or_banks [list data_group1 data_group2]`
- `ResetPowerMapping -group_or_banks data_group`
- `ResetPowerMapping -group_or_banks [list bank1 bank2]`

### Related Commands

[ResetPowerMapping](#)

## ResetPreserveUnusedPinsInBank

Resets the status of the pins of the specified device that are preserved.

### Return

bool

### Syntax

```
ResetPreserveUnusedPinsInBank deviceInstanceName {bankNameList}
```

## Parameters

| Parameter  | Description                                                         | Type        | Optional |
|------------|---------------------------------------------------------------------|-------------|----------|
| deviceName | Specifies the name of the device instance.                          | string      | false    |
| bankNames  | Specifies a list of bank names whose preserved pins is to be reset. | string_list | false    |

## Examples

- `ResetPreserveUnusedPinsInBank U1 {13 14}`
- `ResetPreserveUnusedPinsInBank U1 5`

## Related Commands

[SetPreserveUnusedPinsInBank](#)

# ResetResources

Removes the specified resource information of the device instance, interface instance, or protocol.

## Return

bool

## Syntax

```
ResetResources -d deviceInstanceName [-i interfaceInstanceName]
```

## Parameters

| Parameter | Description                                                                                                                                                      | Type   | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device instance.                                                                                                                       | string | false    |
| -i        | Specifies the name of the interface instance or protocol. In case argument is not specified, the command resets the resources for the specified device instance. | string | true     |

## Examples

```
ResetResources -d U1
```

## Related Commands

- [MapResources](#)
- [GetResources](#)

# ResetSpecifyNetNames

Removes the names of the nets for the specified interface or device protocols.

## Return

bool

## Syntax

```
ResetSpecifyNetNames [interfaceInstanceName/ProtocolName]
```

## Parameters

| Parameter     | Description                                                                                     | Type   | Optional |
|---------------|-------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface instance or protocol name whose net names is to be removed. | string | no       |

## Examples

- `ResetSpecifyNetNames U5`
- `ResetSpecifyNetNames U1_U2`

## Related Commands

[SpecifyNetNames](#)

# ResetUsePins

Removes the values of the Use Pins column for the specified interface instance.

## Return

bool

## Syntax

```
ResetUsePins interfaceInstanceName
```

## Parameters

| Parameter     | Description                                                                             | Type   | Optional |
|---------------|-----------------------------------------------------------------------------------------|--------|----------|
| interfaceName | Specifies the name of the interface instance of which use pins values is to be removed. | string | no       |

## Examples

```
ResetUsePins U2
```

## Related Commands

[ImportConstraints](#)

# RotateInstance

Rotates the specified instance to the specified angle.

## Return

bool

## Syntax

```
RotateInstance {instance_name_list} degree
```

## Parameters

| Parameter     | Description                                       | Type        | Optional |
|---------------|---------------------------------------------------|-------------|----------|
| instance_name | list of the instance names that is to be rotated. | string_list | false    |
| degree        | Specifies the rotation degree.                    | double      | false    |

## Examples

- `RotateInstance XP1 10`
- `RotateInstance U1 10`
- `RotateInstance [list U1 U2 U3 U4] 45`

## Related Commands

- [GetInstanceRotation](#)
- [UpdateInstanceLocation](#)
- [MoveInstance](#)
- [FlipInstance](#)
- [InitInstancePinLocation](#)

# RunDesign

Runs the design and creates connections between the instances.

## Return

int

## Syntax

```
RunDesign
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
RunDesign
```

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesignWithRunSet](#)
- [RunInstance](#)
- [LoadProcessOptions](#)
- [SaveProcessOptions](#)

## RunDesignWithRunSet

Runs the design with saved process options. Note: Execution is based on specific run option order.

### Return

bool

### Syntax

```
RunDesignWithRunSet {list_of_run_set}
```

### Parameters

| Parameter | Description                                             | Type        | Optional |
|-----------|---------------------------------------------------------|-------------|----------|
| runset    | Specifies a list of run options that is to be executed. | string_list | false    |

### Examples

- `RunDesignWithRunSet [GetAllProcessOptionNames]`
- `RunDesignWithRunSet {run_protocol run_memory run_virtual_interfaces}`

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesign](#)
- [RunInstance](#)
- [LoadProcessOptions](#)
- [SaveProcessOptions](#)

## RunInstance

Processes the specified instance with the present set of process options to create connections.

### Return

int

### Syntax

```
RunInstance instance_name
```

## Parameters

| Parameter     | Description                                           | Type   | Optional |
|---------------|-------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance that is to be run. | string | false    |

## Examples

- `RunInstance U1`
- `RunInstance XP2`

## Related Commands

- [RunDesign](#)
- [LoadProcessOptions](#)
- [RunDesignWithRunSet](#)
- [SaveProcessOptions](#)

# SaveProcessOptions

Exports the current process options to the file `$project_dir/$process_option_name.xml`. The process options must be defined earlier using the Process Option Editor.

## Return

bool

## Syntax

```
SaveProcessOptions processOptionTagName
```

## Parameters

| Parameter            | Description                                                                                                                             | Type   | Optional |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| processOptionTagName | Specifies the process option name, created in the Process Option Editor. The specified process option is used as a file name to export. | string | false    |

## Examples

- `SaveProcessOptions run_protocol`
- `SaveProcessOptions run_memory`

## Related Commands

- [GetAllProcessOptionNames](#)
- [RunDesign](#)
- [RunInstance](#)
- [RunDesignWithRunSet](#)
- [LoadProcessOptions](#)
- [RemoveProcessOption](#)
- [RemoveAllProcessOptions](#)
- [RenameProcessOption](#)

## SaveProject

Saves the design settings of the current project.

### Return

bool

### Syntax

```
SaveProject
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
SaveProject
```

## Related Commands

- [SaveProjectAs](#)
- [GetProjectFilePath](#)
- [OpenProject](#)
- [CloseProject](#)

## SaveProjectAs

Creates a copy of the current project in the specified path.

### Return

bool

### Syntax

```
SaveProjectAs newProjAbsolutePath
```

## Parameters

| Parameter                        | Description                                                                      | Type   | Optional |
|----------------------------------|----------------------------------------------------------------------------------|--------|----------|
| <code>newProjAbsolutePath</code> | Specifies the path and name for the project in which the project is to be saved. | string | false    |

## Examples

```
SaveProjectAs D:/fsp_project/saveas_test.fsp
```

## Related Commands

- [SaveProject](#)
- [GetProjectFilePath](#)
- [OpenProject](#)
- [CloseProject](#)

# SaveWorkFlowAs

Saves the workflow at the specified path.

## Return

bool

## Syntax

```
SaveWorkFlowAs workflow_xml_filepath
```

## Parameters

| Parameter                          | Description                              | Type   | Optional |
|------------------------------------|------------------------------------------|--------|----------|
| <code>workflow_xml_filepath</code> | Path were the workflow file is to saved. | string | false    |

## Examples

```
SaveWorkFlowAs D:/workflow/new_workflow.xml
```

## Related Commands

[LoadWorkFlow](#)

# SetAllegroCPMFileName

Points the DE HDL project file (.cpm) to the specified CPM file.

## Return

bool



## Syntax

```
SetAllegroCPMFileName cpm_file_path generate_symbol_libray_name
```

## Parameters

| Parameter                   | Description                                                                | Type   | Optional |
|-----------------------------|----------------------------------------------------------------------------|--------|----------|
| cpm_file_path               | Specifies the path and name of the DE HDL project file.                    | string | false    |
| generate_symbol_libray_name | Specifies the name of the library in which the symbols is to be generated. | string | false    |

## Examples

```
SetAllegroCPMFileName C:/SPB_Data/fsp_working/tcl_test_setup/tcl_command.cpm fsp_fe_lib
```

## Related Commands

- [GetAllegroCPMFileName](#)
- [GetGenerateSymbolLibraryName](#)
- [SetGenerateSymbolLibraryName](#)

# SetAutoNetGroupInterface

Enables the auto-create NetGroups flag for the interfaces. When enabled, the NetGroups for the interfaces are automatically created at the time of instantiation.

## Return

void

## Syntax

```
SetAutoNetGroupInterface is_auto_net_group
```

## Parameters

| Parameter         | Description                                                                                                   | Type | Optional |
|-------------------|---------------------------------------------------------------------------------------------------------------|------|----------|
| is_auto_net_group | Specifies whether the auto-create net groups flag is to be set or reset. The valid values are true and false. | bool | false    |

## Examples

```
SetAutoNetGroupInterface true
```

## Related Commands

- [SetAutoNetGroupProtocol](#)
- [AutoNetGroupDesignGroupWise](#)
- [SetMaximumNetGroupSize](#)

## SetAutoNetGroupProtocol

Enables the auto-create NetGroups for protocols option. When enabled the NetGroups are automatically defined during protocol creation.

### Return

void

### Syntax

```
SetAutoNetGroupProtocol is_auto_net_group
```

### Parameters

| Parameter         | Description                                                                                  | Type | Optional |
|-------------------|----------------------------------------------------------------------------------------------|------|----------|
| is_auto_net_group | Specifies whether the setting is to be enabled or disabled. Valid values are true and false. | bool | false    |

### Examples

```
SetAutoNetGroupProtocol true
```

### Related Commands

- [SetAutoNetGroupInterface](#)
- [AutoNetGroupDesignGroupWise](#)
- [SetMaximumNetGroupSize](#)

## SetBoardDimensionUnits

Sets the design units.

### Return

bool

### Syntax

```
SetBoardDimensionUnits dimension_unit (mm|cm|meter|mils|inch|micron|mili)
```

### Parameters

| Parameter      | Description                                                                                                      | Type   | Optional |
|----------------|------------------------------------------------------------------------------------------------------------------|--------|----------|
| dimension_unit | Specifies the dimension units. The value must be entered in string. Valid values are inch, cm, micron, mils, mm. | string | false    |

### Examples

- ```
SetBoardDimensionUnits inch
```
- ```
SetBoardDimensionUnits cm
```

## Related Commands

- [GetBoardHeight](#)
- [GetBoardWidth](#)
- [SetBoardHeight](#)
- [SetBoardWidth](#)
- [GetBoardDimensionUnits](#)
- [GetPinXCoordinate](#)
- [GetPinYCoordinate](#)
- [GetInstanceXCoordinate](#)
- [GetInstanceYCoordinate](#)
- [GetInstanceHeight](#)
- [GetInstanceWidth](#)

## SetBoardHeight

Sets the board height in current design unit.

### Return

bool

### Syntax

```
SetBoardHeight board_height_value
```

### Parameters

| Parameter          | Description                              | Type   | Optional |
|--------------------|------------------------------------------|--------|----------|
| board_height_value | Specifies the value of the board height. | double | false    |

### Examples

- ```
SetBoardHeight 10
```
- ```
SetBoardHeight 15
```

## Related Commands

- [GetBoardHeight](#)
- [GetBoardWidth](#)
- [SetBoardWidth](#)
- [GetBoardDimensionUnits](#)

## SetBoardWidth

Set the board width in current design unit.

## Return

bool

## Syntax

```
SetBoardWidth board_width_value
```

## Parameters

| Parameter         | Description                                               | Type   | Optional |
|-------------------|-----------------------------------------------------------|--------|----------|
| board_width_value | Specifies the value of the board width that is to be set. | double | false    |

## Examples

- `SetBoardWidth 10`
- `SetBoardWidth 15`

## Related Commands

- [GetBoardHeight](#)
- [GetBoardWidth](#)
- [SetBoardHeight](#)
- [GetBoardDimensionUnits](#)

# SetCachedNetNames

Sets the cached names for the nets of interface or protocol. The cached net names can be used later as generated net name while synthesizing the design.

## Return

bool

## Syntax

```
SetCachedNetNames [-i interface_or_protocol_or_vi_name] -pins comma_separated_pin_names -ports comma_separated_net_names
```

## Parameters

| Parameter | Description                                                                                                                                  | Type   | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or device protocol.                                                                             | string | false    |
| -pins     | Specifies a list of pin names or signal names of the specified interface instance or protocol for which the cached names is to be specified. | string | false    |
| -ports    | Specifies a net names in comma separated format that is to be assigned on the nets of the specified interface instance or protocol pins.     | string | false    |

## Examples

- `SetCachedNetNames -i XP1 -pins {DDR3_A0,DDR3_A1,DDR3_A2} -ports {NET_A0,NET_A1,NET_A2}`
- `SetCachedNetNames -i U1_U2_U3 -pins {DDR3_DQ0,DDR3_DQ1,DDR3_DQ2} -ports {DATA_DQ0,DATA_DQ1,DATA_DQ2}`

## Related Commands

[ResetCachedNetNames](#)

# SetCaptureINIFilePath

Sets the capture.ini file (OrCAD settings) for the current design. The settings stored in the capture.ini file is used during generating OrCAD schematics.

## Return

void

## Syntax

```
SetCaptureINIFilePath ini_file_name
```

## Parameters

| Parameter     | Description                                                      | Type   | Optional |
|---------------|------------------------------------------------------------------|--------|----------|
| ini_file_name | Specifies the name and path where the config.ini file is stored. | string | false    |

## Examples

```
SetCaptureINIFilePath d:/capture.ini
```

## Related Commands

- [GetCaptureINIFilePath](#)
- [ChangeGlobalOrCADLibraryPath](#)

# SetClockBuffer

Sets the clock buffer type for the specified clock\_region constraint groups.

## Return

bool

## Syntax

```
SetClockBuffer interfaceInstance groupName clockBufferType
```

## Parameters

| Parameter       | Description                                                                            | Type   | Optional |
|-----------------|----------------------------------------------------------------------------------------|--------|----------|
| instance_name   | Specifies the name of the interface instance.                                          | string | false    |
| groupName       | Specifies the name of the clock_region constraint group.                               | string | false    |
| clockBufferName | Specifies the type of clock buffer that is to be used. Valid values are BUFR or BUFIO. | string | false    |

## Examples

- `SetClockBuffer U2 Data_ByteL BUFR`
- `SetClockBuffer U2 Data_ByteL BUFIO`

## Related Commands

[GetClockBuffer](#)

# SetContiguousSignal

Updates the information of the vector signals of the interface group that is to be connected in a contiguous die locations of the FPGA.

## Return

bool

## Syntax

```
SetContiguousSignal interfaceInstanceName groupName {{vector_signal_list}} [isSkipConnectedPins]
```

## Parameters

| Parameter           | Description                                                                                                                                                                                                                                               | Type               | Optional |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|----------|
| instance_name       | Specifies the name of the interface instance.                                                                                                                                                                                                             | string             | false    |
| groupName           | Specifies the name of the group of the specified interface instance.                                                                                                                                                                                      | string             | false    |
| vector_signal_list  | Specifies a list of vector signal names list. For Interleaving vectors list will contain two vector names.                                                                                                                                                | string_string_list | false    |
| isSkipConnectedPins | Specifies the value to skip the already connected pins in contiguous connections. Valid values are true or false. The default value is false, that means you can connect pins to FPGA in a contiguous fashion only when no connected pins are in between. | bool               | true     |

## Examples

- `SetContiguousSignal U2 Address_Control {{A} {BA} {C D}}`
- `SetContiguousSignal U2 data_bytel {{DQ CQ} {DM}} true`

## Related Commands

- [GetAllContiguousSignal](#)
- [GetContiguousSignal](#)

## SetContiguousSignalForProtocol

Updates the information of the vector signals of the interface group that is to be connected in a contiguous die locations of the FPGA in a protocol.

### Return

bool

### Syntax

```
SetContiguousSignalForProtocol protocol_name deviceName groupName {{vector_signal_list}} [isSkipConnectedPins]
```

### Parameters

| Parameter           | Description                                                                                                                                                                                                                                               | Type               | Optional |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|----------|
| protocolName        | Specifies the name of the device protocol.                                                                                                                                                                                                                | string             | false    |
| deviceNameList      | Specifies the name of the device.                                                                                                                                                                                                                         | string             | false    |
| groupName           | Specifies the name of the group of the specified interface instance.                                                                                                                                                                                      | string             | false    |
| vector_signal_list  | Specifies a list of vector signal names list. For Interleaving vectors list will contain two vector names.                                                                                                                                                | string_string_list | false    |
| isSkipConnectedPins | Specifies the value to skip the already connected pins in contiguous connections. Valid values are true or false. The default value is false, that means you can connect pins to FPGA in a contiguous fashion only when no connected pins are in between. | bool               | true     |

### Examples

- `SetContiguousSignalForProtocol U3_U1 U3 Address_Control {{A} {BA} {C D}}`
- `SetContiguousSignalForProtocol U1_U4 U1 data_byte1 {{DQ CQ} {DM}} true`

### Related Commands

- [GetAllContiguousSignal](#)
- [GetContiguousSignal](#)
- [SetContiguousSignal](#)

## SetDehdlFPGAHierBlockLibrayName

Sets the library name for FPGA hierarchal blocks generation.

### Return

void

### Syntax

```
SetDehdlFPGAHierBlockLibrayName library_name
```

## Parameters

| Parameter    | Description                        | Type   | Optional |
|--------------|------------------------------------|--------|----------|
| library_name | Specifies the name of the library. | string | false    |

## Examples

- `SetDehdlFPGAHierBlockLibraryName fsp_fe_lib`
- `SetDehdlFPGAHierBlockLibraryName worklib`

## Related Commands

[GetDehdlFPGAHierBlockLibraryName](#)

# SetDesignConnectivityView

Sets the specified view in Design Connectivity.

## Return

bool

## Syntax

```
SetDesignConnectivityView -n view_name
```

## Parameters

| Parameter | Description                                                                     | Type   | Optional |
|-----------|---------------------------------------------------------------------------------|--------|----------|
| -n        | Specifies the view name. For example, Pin View, Net View, Externs, and Interns. | string | false    |

## Examples

- `SetDesignConnectivityView -n \"Pin View\"`
- `SetDesignConnectivityView -n \"Net View\"`
- `SetDesignConnectivityView -n \"Targeted Pin View\"`
- `SetDesignConnectivityView -n \"Externs and Interns\"`

## Related Commands

- [ExportCSVfromDesignConnectivity](#)
- [ImportCSVInDesignConnectivity](#)

# SetDesignExplorerView

Sets the specified view in Design Connectivity.



## Return

bool

## Syntax

```
SetDesignExplorerView -n view_name
```

## Parameters

| Parameter | Description                                                                     | Type   | Optional |
|-----------|---------------------------------------------------------------------------------|--------|----------|
| -n        | Specifies the view name. For example, Pin View, Net View, Externs, and Interns. | string | no       |

## Examples

- `SetDesignExplorerView -n "Pin View"`
- `SetDesignExplorerView -n "Net View"`
- `SetDesignExplorerView -n "Externs and Interns"`

## Related Commands

- [ExportCSVfromDesignExplorer](#)
- [ImportCSVInDesignExplorer](#)

# SetDesignNetNameTemplate

Sets the net name template for the design nets if no net name template is defined earlier. However, if the net name template is already defined, updates the existing net name template with the specified values.

## Return

bool

## Syntax

```
SetDesignNetNameTemplate {template_values_list}
```

## Parameters

| Parameter            | Description                                                                       | Type        | Optional |
|----------------------|-----------------------------------------------------------------------------------|-------------|----------|
| template_values_list | Specifies a list of valid values that is to be defined for the net name template. | string_list | false    |

## Examples

- `SetDesignNetNameTemplate {\ "_\" \"Device Instance Name\" \"_\" \"Interface Instance Name\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Device Instance Name\" \"_\" \"Device Pin Number\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Interface Instance Name\" \"_\" \"Interface Pin Number\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Device Instance Name\" \"_\" \"Interface Instance Name\" \"_\" \"Interface Pin Name\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Device Instance Name\" \"_\" \"Interface Instance Name\" \"_\" \"Device Pin Name\"}`
- `SetDesignNetNameTemplate {\ "_\" \"RTL Port Name\" \"_\" \"IO Standard\" \"_\" \"Target Pin Function\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Group Name\" \"_\" \"Diff Type\" \"_\" \"Direction\"}`
- `SetDesignNetNameTemplate {\ "_\" \"Bit Index\" \"_\" \"Bit Index With Vector Notation\" \"_\" \"Bus or Scalar Name\"}`

## Related Commands

[SetDesignProtocolNetNameTemplate](#)

# SetDesignProtocolNetNameTemplate

Sets the net name template for the device protocols if no net name template is defined earlier. However, if the net name template is already defined, updates the existing net name template with the specified values.

## Return

bool

## Syntax

```
SetDesignProtocolNetNameTemplate {template_values_list}
```

## Parameters

| Parameter            | Description                                                                       | Type        | Optional |
|----------------------|-----------------------------------------------------------------------------------|-------------|----------|
| template_values_list | Specifies a list of valid values that is to be defined for the net name template. | string_list | false    |

## Examples

- `SetDesignProtocolNetNameTemplate {\ "_\" \"Start Instance Name\" \"_\" \"End Instance Name\" \"_\" \"Protocol Signal Name\"}`
- `SetDesignProtocolNetNameTemplate {\ "_\" \"Connected Instances\" \"_\" \"Start Pin Number\"}`
- `SetDesignProtocolNetNameTemplate {\ "_\" \"Protocol Name\" \"_\" \"End Pin Number\"}`
- `SetDesignProtocolNetNameTemplate {\ "_\" \"RTL Port Name\" \"_\" \"IO Standard\"}`
- `SetDesignProtocolNetNameTemplate {\ "_\" \"Group Name\" \"_\" \"Diff Type\"}`
- `SetDesignProtocolNetNameTemplate {\ "_\" \"Bit Index\" \"_\" \"Bit Index With Vector Notation\" \"_\" \"Bus or Scalar Name\"}`

## Related Commands

[SetDesignNetNameTemplate](#)

## SetDesignTopBlockLibAndName

Sets the schematics root library and block.

### Return

void

### Syntax

```
SetDesignTopBlockLibAndName library_name_with_block_name
```

### Parameters

| Parameter                    | Description                                                                                   | Type   | Optional |
|------------------------------|-----------------------------------------------------------------------------------------------|--------|----------|
| library_name_with_block_name | Specifies the names of the library with block. Note: Both the names must be separated by ':'. | string | false    |

### Examples

```
SetDesignTopBlockLibAndName fsp_fe_lib:root
```

### Related Commands

- [SetFPGAHierBlockLibAndName](#)
- [GetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [GetDesignTopBlockLibAndName](#)

## SetDevicePreservePins

Preserves the specified type of pins in all the banks of the device instance.

### Return

bool

### Syntax

```
SetDevicePreservePins device_instance_name pin_type_to_preserve
```

### Parameters

| Parameter            | Description                                                                                                                                                                           | Type   | Optional |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                                                                                                                            | string | false    |
| pin_type_to_preserve | Specifies the type of pins that is to be preserved. Valid values for Xilinx devices {NONE, VREF, VRPVRN, VREF & VRPVRN} and for Altera devices {NONE, RUPRDN, RZQ, VREF, RZQ & VREF}. | string | false    |

### Examples

```
SetDevicePreservePins U1 VREF
```

## Related Commands

- [SetPreservePins](#)
- [GetPreservePins](#)

## SetDoNotCombineDifferentVoltageInputsintoSameBank

Updates the device instance setting which is used to combine or not combine different voltage level input signals into same bank of FPGA. This command is only for Altera FPGAs.

## Return

bool

## Syntax

```
SetDoNotCombineDifferentVoltageInputsintoSameBank device_instance_name flag_value
```

## Parameters

| Parameter            | Description                                                                                                                                                                                      | Type   | Optional |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                                                                                                                                       | string | false    |
| flag_value           | flag_value true will not allow input signals with different voltage value into same bank of FPGA. flag_value false will allow input signals with different voltage value into same bank of FPGA. | bool   | false    |

## Examples

```
SetDoNotCombineDifferentVoltageInputsintoSameBank U1 true
```

## Related Commands

[GetDeviceInstanceList](#)

## SetDoNotConnectPin

Marks the pins as do not connect.

## Return

bool

## Syntax

```
SetDoNotConnectPin instance_name {pin_number_list}
```

## Parameters

| Parameter       | Description                                                                                    | Type        | Optional |
|-----------------|------------------------------------------------------------------------------------------------|-------------|----------|
| instance_name   | Specifies the name of the instance.                                                            | string      | false    |
| pin_number_list | Specified a list of pin numbers of the specified instance that is to be set as do not connect. | string_list | false    |

## Examples

```
SetDoNotConnectPin J1 {1 2 3 4 5 6 7}
```

## Related Commands

- [GetAllDoNotConnect](#)
- [GetDoNotConnect](#)
- [ResetDoNotConnectPin](#)
- [ResetAllDoNotConnect](#)

## SetDontUseBanks

Sets the Don't Use Banks for the specified groups of the interface instance.

## Return

bool

## Syntax

```
SetDontUseBanks interfaceInstanceName {groupNameList} {dontUseBankList}
```

## Parameters

| Parameter             | Description                                                                     | Type        | Optional |
|-----------------------|---------------------------------------------------------------------------------|-------------|----------|
| interfaceInstanceName | Specifies the name of the interface instance.                                   | string      | false    |
| groupNameList         | Specifies the list of group names for which the Use Bank settings is to be set. | string_list | false    |
| bankNameList          | Specifies the list of bank names of the targeted device.                        | string_list | false    |

## Examples

- ```
SetDontUseBanks U2 {Data_ByteL Data_ByteU} {10 11 12}
```
- ```
SetDontUseBanks U2 Address_control 1
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanksForProtocol](#)

## SetDontUseBanksForProtocol

Sets the Don't Use Banks for the groups of the specified device protocol.

### Return

bool

### Syntax

```
SetDontUseBanksForProtocol protocolName {deviceNameList} {groupNameList} {bankNameList}
```

### Parameters

| Parameter      | Description                                                                                   | Type        | Optional |
|----------------|-----------------------------------------------------------------------------------------------|-------------|----------|
| protocolName   | Specifies the name of the device protocol.                                                    | string      | false    |
| deviceNameList | Specifies the list of device names to update dont use banks on the required side of protocol. | string_list | false    |
| groupNameList  | Specifies the list of group names for which the Don't Use Banks setting is to be set.         | string_list | false    |
| bankNameList   | Specifies the list of device bank names.                                                      | string_list | false    |

### Examples

```
SetDontUseBanksForProtocol U3_U1 U3 {Data_ByteL Data_ByteU} {7 8 9}
```

### Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ReOptimizeProtocol](#)

## SetEnvVariable

Set the specified environment variable.

### Return

bool

## Syntax

```
SetEnvVariable envVarName envVarValue
```

## Parameters

| Parameter   | Description                                                                          | Type   | Optional |
|-------------|--------------------------------------------------------------------------------------|--------|----------|
| envVarName  | Specifies the name of the environment variable of which value is required to be set. | string | false    |
| envVarValue | Specifies the value of the environment variable.                                     | string | false    |

## Examples

- `SetEnvVariable MY_ENV 10`
- `SetEnvVariable FSP_WORKING_DIR C:/fsp_working`

## Related Commands

- [GetEnvVariable](#)
- [GetEnvVariables](#)

# SetExternPin

Sets the specified pin as external pin.

## Return

bool

## Syntax

```
SetExternPin
```

## Parameters

| Parameter     | Description                                                            | Type   | Optional |
|---------------|------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                                    | string | false    |
| pin_number    | Specifies the pin number of the pin that is to be set as external pin. | string | false    |
| net_name      | Specifies the name of the net that is to be assigned to external pin.  | string | false    |

## Examples

```
SetExternPin U6 1 test_extern
```

## Related Commands

[ResetExternPin](#)

## SetFPGAHierBlockLibAndName

Sets the specified names as the FSP design block library name and the block name.

### Return

void

### Syntax

```
SetFPGAHierBlockLibAndName instance_name library_name_with_block_name
```

### Parameters

| Parameter                    | Description                                                                                                                                  | Type   | Optional |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| instance_name                | Specifies the name of the instance.                                                                                                          | string | false    |
| library_name_with_block_name | Specifies the name of the FPGA block library name with the block name that is to be set. The names should separated with colon character'.'. | string | false    |

### Examples

```
SetFPGAHierBlockLibAndName U5 fsp_fe_lib:h5_hiersym
```

### Related Commands

- [GetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [GetDesignTopBlockLibAndName](#)

## SetFSPBlockLibAndName

Sets the FSP's design block library name and block name.

### Return

void

### Syntax

```
SetFSPBlockLibAndName library_name_with_block_name
```

### Parameters

| Parameter                    | Description                                                                                                        | Type   | Optional |
|------------------------------|--------------------------------------------------------------------------------------------------------------------|--------|----------|
| library_name_with_block_name | Specifies the name of the FSP design block library with block name. Note: Both the names must be separated by '.'. | string | false    |

### Examples

```
SetFSPBlockLibAndName fsp_fe_lib:test_fsp
```



## Related Commands

- [SetFPGAHierBlockLibAndName](#)
- [GetFPGAHierBlockLibAndName](#)
- [SetDesignTopBlockLibAndName](#)
- [GetFSPBlockLibAndName](#)
- [GetDesignTopBlockLibAndName](#)

## SetGenerateSymbolLibraryName

Sets the specified library name as default symbol library. After setting, the schematics symbol are generated in the specified library directory.

### Return

void

### Syntax

```
SetGenerateSymbolLibraryName library_name
```

### Parameters

| Parameter    | Description                                                                                                                                                                   | Type   | Optional |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| library_name | Specifies the name of the library where to generate the symbols. For DE-HDL, specify the library name and for OrCAD, specify the name and path where to generate the symbols. | string | false    |

### Examples

- `SetGenerateSymbolLibraryName d:/test.olb`
- `SetGenerateSymbolLibraryName fsp_fe_lib`
- `ChangeGlobalOrCADLibraryPath`

## Related Commands

[GetGenerateSymbolLibraryName](#)

## SetIsCheckSymbolLargerThanPageBorder

Enables or disables the check whether the symbol(s) size larger than the schematic page border. This check is used while generating schematics.

### Return

void

### Syntax

```
SetIsCheckSymbolLargerThanPageBorder isCheckSymbolWithPageBorder
```

## Parameters

| Parameter    | Description                                                                           | Type | Optional |
|--------------|---------------------------------------------------------------------------------------|------|----------|
| check_symbol | Specifies whether to enable or disable symbol check. Valid values are true and false. | bool | false    |

## Examples

- `SetIsCheckSymbolLargerThanPageBorder true`
- `SetIsCheckSymbolLargerThanPageBorder false`
- `SetIsCheckSymbolLargerThanPageBorder 0`

## Related Commands

[IsCheckSymbolLargerThanPageBorder](#)

# SetIsFilterLogMessages

Enables skipping or enabling log message dump.

## Return

void

## Syntax

```
SetIsFilterLogMessages is_skip_log_messages
```

## Parameters

| Parameter            | Description                                                                                                                                                                       | Type | Optional |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|----------|
| is_skip_log_messages | Specifies log message to be displayed in log window. The valid values are true and false. True to skip listing log message and false to enable listing log message in log window. | bool | false    |

## Examples

- `SetIsFilterLogMessages true`
- `SetIsFilterLogMessages false`

## Related Commands

# SetIsGenerateSFReport

Enables or disables the generation of report for unconnected signals during synthesis. The Synthesis Failure Report provides you a detail explanation and reason for the failed connections.

## Return

bool

## Syntax

```
SetIsGenerateSFReport isGenerate
```

## Parameters

| Parameter  | Description                                                                                                            | Type | Optional |
|------------|------------------------------------------------------------------------------------------------------------------------|------|----------|
| isGenerate | Specifies the value to enable or disable the Synthesis Failure Report generation. The valid values are true and false. | bool | false    |

## Examples

- `SetIsGenerateSFReport true`
- `SetIsGenerateSFReport false`

## Related Commands

[IsGenerateSFReport](#)

## SetIsolateStatus

Modifies the Isolate High Speed Serial I/O's flag for the specified device instance. This command is supported for Virtex4 or Virtex5 devices.

## Return

bool

## Syntax

```
SetIsolateStatus deviceInstanceName true|false
```

## Parameters

| Parameter          | Description                                                                                                   | Type   | Optional |
|--------------------|---------------------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance for which Isolate High Speed Serial I/O's status is to be modified. | string | false    |
| isIsolate          | Specifies the status. Valid values are true and false.                                                        | bool   | false    |

## Examples

```
SetIsolateStatus U1 true
```

## Related Commands

[GetIsolateStatus](#)

## SetIsPerformSecondPassOptimization

Enables second pass optimization flag in the design. The second pass optimization flag decides whether the synthesis will succeed with a second pass run to minimize the nets crossovers in the design.

## Return

void

## Syntax

```
SetIsPerformSecondPassOptimization is_perform_second_pass_optimization
```

## Parameters

| Parameter                           | Description                                                                                                                                                                     | Type | Optional |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|----------|
| is_perform_second_pass_optimization | Specifies whether second pass optimization is to be performed in the design. The valid values are true and false. True to enable second pass optimization and false to disable. | bool | false    |

## Examples

- `SetIsPerformSecondPassOptimization true`
- `SetIsPerformSecondPassOptimization false`

## Related Commands

- [IsPerformSecondPassOptimization](#)
- [ToggleSecondPassOptimization](#)

# SetLicenseType

Sets the specified license as a current license. For this command to run successfully, you must close all the opened projects and run the command.

## Return

bool

## Syntax

```
SetLicenseType licenceType
```

## Parameters

| Parameter   | Description                              | Type   | Optional |
|-------------|------------------------------------------|--------|----------|
| licenceType | Specifies the string of the FSP license. | string | false    |

## Examples

```
SetLicenseType Allegro_FPGA_System_Plan_GXL
```

## Related Commands

- [GetAvailableLicenses](#)
- [GetAvaliableLicenseType](#)
- [GetCurrentLicenseType](#)

## SetMaximumNetGroupSize

Defines the maximum size of the NetGroups. The specified size will be considered by FSP during auto-creation of NetGroups.

### Return

void

### Syntax

```
SetMaximumNetGroupSize max_net_group_size
```

### Parameters

| Parameter          | Description                                 | Type | Optional |
|--------------------|---------------------------------------------|------|----------|
| max_net_group_size | Specifies the maximum size of the NetGroup. | int  | false    |

### Examples

```
SetMaximumNetGroupSize 128
```

### Related Commands

- [GetMaximumNetGroupSize](#)
- [SetAutoNetGroupInterface](#)
- [SetAutoNetGroupProtocol](#)
- [AutoNetGroupDesignGroupWise](#)

## SetMaxOutputsPerBank

Sets the maximum number of outputs allowed into a bank for the specified device instance.

### Return

bool

### Syntax

```
SetMaxOutputsPerBank deviceInstancename bankname numberofoutputs
```

### Parameters

| Parameter          | Description                                                                              | Type   | Optional |
|--------------------|------------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of device instance.                                                   | string | false    |
| bankName           | Specifies the name of the bank for which the maximum number of outputs is to be defined. | string | false    |
| pinCount           | Specifies the maximum number of output pins that is to be allowed for connections.       | int    | false    |

## Examples

```
SetMaxOutputsPerBank U2 3 10
```

## Related Commands

[GetMaxOutputsPerBank](#)

# SetOutputDirPath

Sets the specified directory path as default output directory. After setting, files such as design netlist, constraints, and more are generated in the output directory.

## Return

bool

## Syntax

```
SetOutputDirPath directory_path
```

## Parameters

| Parameter      | Description                                                              | Type   | Optional |
|----------------|--------------------------------------------------------------------------|--------|----------|
| directory_path | Specifies the directory path that you want to set as a output directory. | string | false    |

## Examples

```
SetOutputDirPath C:/SPB_Data/fsp_working/tcl_test_setup/tcl_command/output
```

## Related Commands

[GetOutputDirPath](#)

# SetPCBOutlineHeight

Updates the PCB outline width in current design unit.

## Return

bool

## Syntax

```
SetPCBOutlineHeight outlineHeight
```

## Parameters

| Parameter     | Description                                   | Type   | Optional |
|---------------|-----------------------------------------------|--------|----------|
| outlineHeight | Specifies the new height for the PCB outline. | double | false    |

## Examples

```
SetPCBOutlineHeight 10.0
```

## Related Commands

- [GetPCBOutlineHeight](#)
- [GetPCBOutlineWidth](#)
- [SetPCBOutlineWidth](#)
- [GetPCBOutlineSettings](#)
- [SetPCBOutlineSettings](#)

## SetPCBOutlineSettings

Sets the width, height, and start point for the PCB outline.

## Return

bool

## Syntax

```
SetPCBOutlineSettings [-ow outline_width] [-oh outline_height] [-x outline_start_x_point] [-y outline_start_y_point]
```

## Parameters

| Parameter | Description                               | Type   | Optional |
|-----------|-------------------------------------------|--------|----------|
| -ow       | Specifies the outline width.              | double | false    |
| -oh       | Specifies the outline height.             | double | false    |
| -x        | Specifies outline start point x location. | double | false    |
| -y        | Specifies outline start point y location. | double | false    |

## Examples

```
SetPCBOutlineSettings -ow 3 -oh 3 -x 1 -y 1
```

## Related Commands

- [GetPCBOutlineHeight](#)
- [SetPCBOutlineHeight](#)
- [SetPCBOutlineWidth](#)
- [GetPCBOutlineWidth](#)
- [GetPCBOutlineSettings](#)

## SetPCBOutlineWidth

Updates the PCB outline width in current design unit.

## Return

bool

## Syntax

```
SetPCBOutlineWidth outlineWidth
```

## Parameters

| Parameter    | Description                                  | Type   | Optional |
|--------------|----------------------------------------------|--------|----------|
| outlineWidth | Specifies the new width for the PCB outline. | double | false    |

## Examples

```
SetPCBOutlineWidth 10.0
```

## Related Commands

- [GetPCBOutlineHeight](#)
- [GetPCBOutlineWidth](#)
- [SetPCBOutlineHeight](#)
- [GetPCBOutlineSettings](#)
- [SetPCBOutlineSettings](#)

# SetPinNameASNetName

Sets or resets the pin name to net name for the specified device instance.

## Return

bool

## Syntax

```
SetPinNameASNetName instance_name value
```

## Parameters

| Parameter     | Description                                                                                           | Type   | Optional |
|---------------|-------------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the device instance of which you want to reset or set the pin name to net name. | string | no       |
| value         | Specifies the value of a flag. The valid values are true and false.                                   | bool   | no       |

## Examples

- ```
SetPinNameASNetName U1 true
```
- ```
SetPinNameASNetName U2 false
```



## Related Commands

[IsPinNameASNetName](#)

## SetPowerRegulator

Maps regulator to a specified instance pin.

### Return

string

### Syntax

```
SetPowerRegulator instance_name pin_number regulator_name
```

### Parameters

| Parameter      | Description                                                                                | Type   | Optional |
|----------------|--------------------------------------------------------------------------------------------|--------|----------|
| instance_name  | Name of the instance.                                                                      | string | false    |
| pin_number     | Specifies the pin number to which the power regulator is to be connected.                  | string | false    |
| regulator_name | Specifies the regulator name. Make sure that the power regulator is defined in the design. | string | false    |

### Examples

- `SetPowerRegulator U3 G8 GND`
- `SetPowerRegulator XP2 168 V_2_`

## Related Commands

- [GetPowerRegulator](#)
- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)
- [SetPowerRegulatorVoltage](#)

## SetPowerRegulatorVoltage

Updates the voltage value of the specified power regulator. To run this command, the specified power regulator must be already defined in the design.

### Return

bool

## Syntax

```
SetPowerRegulatorVoltage regulator_name voltage_value
```

## Parameters

| Parameter      | Description                                                                        | Type   | Optional |
|----------------|------------------------------------------------------------------------------------|--------|----------|
| regulator_name | Name of the existing power regulator for which the voltage value is to be updated. | string | false    |
| voltage_value  | New voltage value that is to be updated.                                           | float  | false    |

## Examples

```
SetPowerRegulatorVoltage V_0_9 0.9
```

## Related Commands

- [AutoAddPowerRegulators](#)
- [AddPowerRegulator](#)
- [DeletePowerRegulator](#)
- [DeletePowerRegulators](#)
- [GetPowerRegulator](#)
- [SetPowerRegulator](#)
- [GetPowerRegulators](#)
- [RenamePowerRegulator](#)
- [GetPowerRegulatorVoltage](#)

## SetPreservePins

Preserves the specified type of pins in the bank of the device instance.

## Return

bool

## Syntax

```
SetPreservePins device_instance_name bank_name pin_type_to_preserve
```

## Parameters

| Parameter            | Description                                                                                                                                                                           | Type   | Optional |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| device_instance_name | Specifies the name of the device instance.                                                                                                                                            | string | false    |
| bank_name            | Specifies the name of the bank.                                                                                                                                                       | string | false    |
| pin_type_to_preserve | Specifies the type of pins that is to be preserved. Valid values for Xilinx devices {NONE, VREF, VRPVRN, VREF & VRPVRN} and for Altera devices {NONE, RUPRDN, RZQ, VREF, RZQ & VREF}. | string | false    |

## Examples

- `SetPreservePins U2 3 VREF`
- `SetPreservePins U2 4 VRPVRN`

## Related Commands

- [SetDevicePreservePins](#)
- [GetPreservePins](#)

# SetPreserveUnusedPinsInBank

Preserves the pins of the specified bank that are not connected.

## Return

bool

## Syntax

```
SetPreserveUnusedPinsInBank deviceInstanceName {bankNameList}
```

## Parameters

| Parameter  | Description                                                                   | Type        | Optional |
|------------|-------------------------------------------------------------------------------|-------------|----------|
| deviceName | Specifies the name of the device instance whose pins is to be preserved.      | string      | false    |
| bankNames  | Specifies a list of the bank names whose unconnected pins is to be preserved. | string_list | false    |

## Examples

- `SetPreserveUnusedPinsInBank U1 {9 10}`
- `SetPreserveUnusedPinsInBank U1 22`

## Related Commands

[ResetPreserveUnusedPinsInBank](#)

# SetRulesWorkingDir

Sets the specified directory path as a rules working directory. For this command to work properly, the specified working directory must be listed in the rules path.

## Return

bool

## Syntax

```
SetRulesWorkingDir rules_dir_path
```

## Parameters

| Parameter      | Description                                                            | Type   | Optional |
|----------------|------------------------------------------------------------------------|--------|----------|
| rules_dir_path | Specifies the rules files path that is to be set as working directory. | string | false    |

## Examples

```
SetRulesWorkingDir \"D:/fsp_working/my_rules\"
```

## Related Commands

- [GetRuleFilePath](#)
- [GetRulesFilePaths](#)
- [RemoveRulesFilePath](#)
- [GetRulesWorkingDir](#)
- [ReportAllRulesFiles](#)
- [GetResolvedRulesFilePaths](#)
- [CheckDesignConsistency](#)

## SetSchematicBoardFileName

Use this command to specify a name for Allegro board file (.brd).

## Return

bool

## Syntax

```
SetSchematicBoardFileName board_file_name
```

## Parameters

| Parameter       | Description                                                              | Type   | Optional |
|-----------------|--------------------------------------------------------------------------|--------|----------|
| board_file_name | Specifies the name that you want to assign to Allegro board file (.brd). | string | false    |

## Examples

```
SetSchematicBoardFileName fsp_proj.brd
```

## Related Commands

- [SetSchematicBoardFilePath](#)
- [GetSchematicBoardFilePath](#)
- [GetSchematicBoardFileName](#)

## SetSchematicBoardFilePath

Use this command to set the directory path for generating Allegro board file (.brd).

## Return

bool

## Syntax

```
SetSchematicBoardFilePath file_path
```

## Parameters

| Parameter | Description                                                                           | Type   | Optional |
|-----------|---------------------------------------------------------------------------------------|--------|----------|
| file_path | Specifies the path to the directory where you want to generate the board file (.brd). | string | false    |

## Examples

```
SetSchematicBoardFilePath D:/physical
```

## Related Commands

- [GetSchematicBoardFilePath](#)
- [GetSchematicBoardFileName](#)
- [SetSchematicBoardFileName](#)

# SetSearchNReplaceNetNamePattern

Replaces the pattern of the existing net names with the specified pattern.

## Return

bool

## Parameters

| Parameter | Description | Type   | Optional |
|-----------|-------------|--------|----------|
| flagValue | -           | string | false    |

## Examples

No Examples

## Related Commands

- [AddSearchNReplaceNetNamePattern](#)
- [RemoveSearchNReplaceNetNamePatterns](#)
- [GetSearchNReplaceNetNamePattern](#)

# SetSkipConnectedPins

Sets the flag skip connected pins at group level. If the flag is on bus signals are connected contiguously by skipping the connected signals in between the bus signals.

## Return

bool

## Syntax

```
SetSkipConnectedPins instance_name group_name is_skip
```

## Parameters

| Parameter     | Description                                   | Type   | Optional |
|---------------|-----------------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface instance. | string | false    |
| group_name    | Specifies the name of the group.              | string | false    |
| is_skip       | Specifies the bool value.                     | bool   | false    |

## Examples

```
SetSkipConnectedPins U1 data_bytel true
```

## Related Commands

- [IsSkipConnectedPinsInContiguousConnections](#)
- [IsSkipConnectedPinsInContiguousConnectionsForProtocol](#)

# SetSnapShotTimeInterval

Sets the snapshot interval for the current design. The command saves the design copy at \$APPDATA/Cadence/\$process\_id/SnapShot/\$project\_name/\$time\_stamp.fsp path on Windows and \$HOME/.config/Cadence/\$process\_id/SnapShot/\$project\_name/\$time\_stamp.fsp path on Linux at specified time interval.

## Return

bool

## Syntax

```
SetSnapShotTimeInterval interval_in_minutes(0:30)
```

## Parameters

| Parameter | Description                                                                                                                                             | Type | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------|------|----------|
| interval  | Specifies the time interval in minutes, at which you want to take the snapshots of the current design. The value should be in range of 0 to 30 minutes. | int  | false    |

## Examples

```
SetSnapShotTimeInterval 5
```

## Related Commands

[GetSnapShotTimeInterval](#)

## SetSymbolPinDirection

Sets the pin directions for the schematics symbol of specified instance. This command can be used only for instance for which symbols generation can be done from FSP.

### Return

bool

### Syntax

```
SetSymbolPinDirection -i instance_name [-d] [-in input_pin_direction] [-out output_pin_direction] [-bi bidirection_pin_direction] [-nc no_connect_pin_direction] [-bp bank_power_pin_direction] [-gp global_power_pin_direction]
```

### Parameters

| Parameter | Description                                                                                                                                                                         | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the instance for which the symbol pin directions is to be set.                                                                                                | string | false    |
| -d        | Specifies whether the pin directions is to be set according to the default directions specified in the ini file.                                                                    | bool   | true     |
| -in       | Specifies the input pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings.        | string | true     |
| -out      | Specifies the output pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings.       | string | true     |
| -bi       | Specifies the inout pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings.        | string | true     |
| -nc       | Specifies the no connect pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings.   | string | true     |
| -bp       | Specifies the bank power pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings.   | string | true     |
| -gp       | Specifies the global power pins direction. Valid values are Top, Bottom, Left, Right, LeftAndRight, and Distribute. Default value is considered as per symbol or ini file settings. | string | true     |

### Examples

- `SetSymbolPinDirection -i U5 -d`
- `SetSymbolPinDirection -i U5 -in Top -out Left -nc Right -bp Left -gp Top`

### Related Commands

[GetInstanceNameList](#)

## SetUseBanks

Sets the use banks for the specified groups. For this command to work properly, ensure that the specified instance is targeted to the device instance.

### Return

bool

## Syntax

```
SetUseBanks interfaceInstanceName {groupNameList} {bankNameList}
```

## Parameters

| Parameter             | Description                                                                    | Type        | Optional |
|-----------------------|--------------------------------------------------------------------------------|-------------|----------|
| interfaceInstanceName | Specifies the name of the interface instance.                                  | string      | false    |
| groupNameList         | Specifies the list of group names for which the Use Bank setting is to be set. | string_list | false    |
| bankNameList          | Specifies the list of bank names of the targeted device.                       | string_list | false    |

## Examples

```
SetUseBanks U2 {Address_Control Data_ByteL} {4 5 6}
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

## SetUseBanksForMultiDevices

Updates the Use Banks settings for the specified group that is targeted to the multiple tester connectors.

## Return

bool

## Syntax

```
SetUseBanksForMultiDevices interfaceInstanceName groupNameList useBanksmap
```

## Parameters

| Parameter     | Description                                                                                                  | Type              | Optional |
|---------------|--------------------------------------------------------------------------------------------------------------|-------------------|----------|
| instance_name | Specifies the name of the interface instance for which the Use Banks setting is to be updated.               | string            | false    |
| groupName     | Specifies a list of group names of the specified interface for which the Use Banks setting is to be updated. | string_list       | false    |
| useBanks      | Specifies the list of bank names that is to be set for the specified group.                                  | string_string_map | false    |



## Examples

- `SetUseBanksForMultiDevices U1 GDDR_TD_G1 {J1 \"bank1\" J2 \"bank1\"}`
- `SetUseBanksForMultiDevices U5 GDDR_SC2_G1 {J3 \"bank1 bank2\" J4 \"bank4 bank5\"}`
- `SetUseBanksForMultiDevices DUT group3 [list MUX1 [list inputs] MUX2 [list inputs]]`
- `SetUseBanksForMultiDevices DUT [list group3:9] [list MUX1 [list inputs] MUX2 [list inputs]]`
- `SetUseBanksForMultiDevices DUT [list group3:9] [list MUX* [list inputs]]`

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

## SetUseBanksForProtocol

Sets the Use Banks for the groups of the specified device protocol.

### Return

bool

### Syntax

```
SetUseBanksForProtocol protocolName {deviceNameList} {groupNameList} {bankNameList}
```

### Parameters

| Parameter      | Description                                                                      | Type        | Optional |
|----------------|----------------------------------------------------------------------------------|-------------|----------|
| protocolName   | Specifies the name of the device protocol.                                       | string      | false    |
| deviceNameList | Specifies the name of the device.                                                | string_list | false    |
| groupNameList  | Specifies the list of group names for which the Use Banks settings is to be set. | string_list | false    |
| bankNameList   | Specifies the list of device bank names that needs be targeted.                  | string_list | false    |

## Examples

```
SetUseBanksForProtocol U3_U1 {U3 U1} {Data_ByteL Data_ByteU} {7 8 9}
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)
- [GetProtocolNames](#)
- [RenameProtocol](#)
- [ReOptimizeProtocol](#)

## SetVoltageForMultiVoltagePins

Applies the specified voltages to the pins that are having multiple voltages.

### Return

bool

### Syntax

```
SetVoltageForMultiVoltagePins device_inst_name pin_name_voltage_map
```

### Parameters

| Parameter            | Description                                 | Type              | Optional |
|----------------------|---------------------------------------------|-------------------|----------|
| device_instance_name | Specifies the name of the device instance.  | string            | false    |
| pin_name_voltage_map | Specifies the voltage required for the pin. | string_string_map | false    |

### Examples

```
SetVoltageForMultiVoltagePins U1 {VCCAUX_IO_G0 1.8 VCCAUX_IO_G1 2 VCCINT 0.9}
```

### Related Commands

[GetDeviceInstanceList](#)

## SpecifyDiffPinTermination

Defines the differential pair termination.

### Return

bool

## Syntax

```
SpecifyDiffPinTermination termination_name diff_pair_termination_name
```

## Parameters

| Parameter                  | Description                                                                                     | Type   | Optional |
|----------------------------|-------------------------------------------------------------------------------------------------|--------|----------|
| termination_name           | Specifies the name of the termination for which the differential pair termination is to be set. | string | false    |
| diff_pair_termination_name | Specifies the name to be used for the differential end termination.                             | string | false    |

## Examples

```
SpecifyDiffPinTermination ser_term ser_term
```

## Related Commands

- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)

## SpecifyNetNames

Assigns the names for the nets of the specified interface or device protocols.

## Return

bool

## Syntax

```
SpecifyNetNames -i [interfaceInstanceName/ProtocolName] -pins (comma_separated_pin_names) -ports (comma_separated_net_names)
```

## Parameters

| Parameter | Description                                                                                                                               | Type   | Optional |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or device protocol.                                                                          | string | no       |
| -pins     | Indicates a list of pin names or signal names of the specified interface instance or protocol for which the net names is to be specified. | string | no       |
| -ports    | Indicates a list of net names that is to be assigned on the nets of the specified interface instance or protocol pins.                    | string | no       |

## Examples

No Examples

## Related Commands

[ResetSpecifyNetNames](#)

# SpecifyOtherEndTermination

Sets other/FPGA end termination

## Return

bool

## Syntax

```
SpecifyOtherEndTermination
```

## Parameters

| Parameter                  | Description                                                            | Type   | Optional |
|----------------------------|------------------------------------------------------------------------|--------|----------|
| termination_name           | Termination name for which differential pair termination has to be set | string | false    |
| other_end_termination_name | Other end termination name                                             | string | false    |

## Examples

```
SpecifyOtherEndTermination ser_term ser_term
```

## Related Commands

- [SpecifyDiffPinTermination](#)
- [AddTermination](#)
- [DeleteTermination](#)
- [MapPowerFilterToInstancePin](#)
- [MapTerminationToInstancePinOtherEnd](#)
- [MapTermination](#)
- [MapTerminationOrCAD](#)

# SplitSymbolBankWise

Splits the symbol according to the banks of the specified instance. For example, creates one symbol per bank.

## Return

bool

## Syntax

```
SplitSymbolBankWise instance_name
```

## Parameters

| Parameter     | Description                                                         | Type   | Optional |
|---------------|---------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance for which symbol is to be split. | string | false    |

## Examples

```
SplitSymbolBankWise U2
```

## Related Commands

- [SplitSymbolConnectionWise](#)
- [MergeAllSymbolSplits](#)

# SplitSymbolConnectionWise

Splits the symbol based on the connections. For example, creates one split for each set of nets that is connected to the specified instance or protocol.

## Return

bool

## Syntax

```
SplitSymbolConnectionWise instance_name
```

## Parameters

| Parameter     | Description                                                             | Type   | Optional |
|---------------|-------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance for which the symbol is to be split. | string | false    |

## Examples

```
SplitSymbolConnectionWise U2
```

## Related Commands

- [SplitSymbolBankWise](#)
- [MergeAllSymbolSplits](#)

# StringListTest

Tests the string list variable type.

## Return

string\_list

## Syntax

```
StringListTest arg
```

## Parameters

| Parameter | Description                                                    | Type        | Optional |
|-----------|----------------------------------------------------------------|-------------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | string_list | false    |

## Examples

- `StringListTest [list string1 string2 string3]`
- `StringListTest {"hello\" \"TCL\"}`
- `StringListTest {test this}`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

## StringStringListTest

Tests the string string list variable type.

## Return

string\_string\_list

## Syntax

```
StringStringListTest arg
```

## Parameters

| Parameter | Description                                                    | Type               | Optional |
|-----------|----------------------------------------------------------------|--------------------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | string_string_list | false    |

## Examples

```
StringStringListTest \"this is string string\" \"list\"
```

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringMapTest](#)

## StringStringMapTest

Tests the specified variable type.

### Return

string\_string\_map

### Syntax

```
StringStringMapTest arg
```

### Parameters

| Parameter | Description                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------|-------------------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | string_string_map | false    |

### Examples

- `StringStringMapTest \"this is example stringstringmap\"`
- `StringStringMapTest [list key1 [list value11 value12] key2 [list value21 value22]]`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringTest](#)
- [StringListTest](#)
- [StringStringListTest](#)

## StringTest

Tests the string variable type.

## Return

string

## Syntax

```
StringTest arg
```

## Parameters

| Parameter | Description                                                    | Type   | Optional |
|-----------|----------------------------------------------------------------|--------|----------|
| arg       | Specifies the value of the variable type that is to be tested. | string | false    |

## Examples

- `StringTest stringtest`
- `StringTest \"Hello TCL\"`

## Related Commands

- [BoolTest](#)
- [IntTest](#)
- [IntListTest](#)
- [DoubleTest](#)
- [DoubleListTest](#)
- [StringListTest](#)
- [StringStringListTest](#)
- [StringStringMapTest](#)

## SwapGroups

Swaps the interface groups.

## Return

bool

## Syntax

```
SwapGroups -d device_name -si source_interface_name -sg source_interface_group_name -di destination_interface_name -dg destination_interface_group_name
```



## Parameters

| Parameter | Description                                            | Type   | Optional |
|-----------|--------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device.                      | string | false    |
| -si       | Specifies the source interface.                        | string | false    |
| -sg       | Specifies the name of the source interface group.      | string | false    |
| -di       | Specifies the name of the destination interface.       | string | false    |
| -dg       | Specifies the name of the destination interface group. | string | false    |

## Examples

```
SwapGroups -d U1 -si XP1 -sg Address_control -di XP1 -dg Data_Input
```

## Related Commands

[GetInstanceNameList](#)

# TargetDevice

Sets the target for the specified group of the instance.

## Return

bool

## Syntax

```
TargetDevice -i interface_instance_name [-g group_name] -d target_device_instance_name
```

## Parameters

| Parameter | Description                                                                  | Type   | Optional |
|-----------|------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface for which the target is to be set.       | string | false    |
| -g        | Specifies the name of the group.                                             | string | true     |
| -d        | Specifies the device instance name to which the interface is to be targeted. | string | false    |

## Examples

- `TargetDevice -i U2 -g Address_Control -d U1`
- `TargetDevice -i U3 -d U1`

## Related Commands

- [GetAllTargetDevice](#)
- [GetTargetDevice](#)
- [GetTargetInstanceListForDevice](#)
- [TargetToMultipleDevices](#)

## TargetToMultipleDevices

Sets the multiple target devices for the specified interface group to a connector. This command is useful to target an interface to multiple TesterConnectors.

### Return

bool

### Syntax

```
TargetToMultipleDevices -i interface_instance_name [-g groupName] -t {target_device_list} [-d daisy_chain]
```

### Parameters

| Parameter | Description                                                                                                                                                                        | Type               | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|----------|
| -i        | Specifies the name of the interface instance.                                                                                                                                      | string             | false    |
| -g        | Specifies the name of the group of the specified interface that is to be connected to multiple connectors. In case not specified, FSP will target all group of interface instance. | string             | true     |
| -t        | Specifies a list of the target connector names to which the specified interface group is to be connected.                                                                          | string_string_list | false    |
| -d        | Specifies whether the group should be connected in daisy chain. If this argument is not specified, FSP will create daisy chain protocol by default.                                | bool               | true     |

### Examples

```
TargetToMultipleDevices -i U1 -g TD_group -t [list [list J1 J2 J3] [list J4 J5 J6]]
```

### Related Commands

- [GetAllTargetDevice](#)
- [GetTargetDevice](#)
- [GetTargetInstanceListForDevice](#)
- [TargetDevice](#)

## ToggleOptimizeTDConnectorUtilization

Toggles the Optimize TC Connector Utilization flag. When this flag is set, the FSP synthesis engine optimally spreads out the assigned pins across all the available TC connectors. If this flag is turned off, the FSP synthesis engine tries to accommodate connections in the minimum possible number of connectors.

### Return

void

### Syntax

```
ToggleOptimizeTDConnectorUtilization
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ToggleOptimizeTDConnectorUtilization
```

## Related Commands

[IsOptimizeTDConnectorUtilization](#)

# ToggleSecondPassOptimization

Toggles the second pass optimization flag. The second pass optimization flag decides whether the synthesis will succeed with a second pass run to minimize the nets crossovers in the design.

## Return

void

## Syntax

```
ToggleSecondPassOptimization
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ToggleSecondPassOptimization
```

## Related Commands

- [SetIsPerformSecondPassOptimization](#)
- [IsPerformSecondPassOptimization](#)

# UnLockInstanceNets

Unlocks all the locked nets that are connected to the specified instance.

## Return

bool

## Syntax

```
UnLockInstanceNets instance_name
```

## Parameters

| Parameter     | Description                                                                     | Type   | Optional |
|---------------|---------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance of which the locked nets you want to unlock. | string | no       |

## Examples

```
UnLockInstanceNets XP1
```

## Related Commands

[LockInstanceNets](#)

## UnlockNets

Unlock nets in design. Specify at least one command option.

## Return

bool

## Syntax

```
UnlockNets [-all] [-clocks] [-constraint_pins] [-nets {list_of_net_names}] [-bus bus_name] [-inst_or_protocol instance_or_protocol_name] [-inst_or_protocol_group_or_bank group_or_bank_name]
```

## Parameters

| Parameter                       | Description                                                                                                | Type        | Optional |
|---------------------------------|------------------------------------------------------------------------------------------------------------|-------------|----------|
| -all                            | Unlocks all the nets.                                                                                      | bool        | true     |
| -clocks                         | Unlocks all the clock nets.                                                                                | bool        | true     |
| -constraint_pins                | Unlocks all the constraint pin nets.                                                                       | bool        | true     |
| -nets                           | Unlocks all the specified nets.                                                                            | string_list | true     |
| -bus                            | Unlocks all the nets that belongs to the specified bus name/                                               | string      | true     |
| -inst_or_protocol               | Unlocks all the nets that belongs to the specified interface or protocol.                                  | string      | true     |
| -inst_or_protocol_group_or_bank | Unlocks all the nets that belongs to the specified group (or bank) of the specified interface or protocol. | string      | true     |

## Examples

- `UnlockNets -all`
- `UnlockNets -clocks`
- `UnlockNets -clocks -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `UnlockNets -clocks -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `UnlockNets -clocks -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `UnlockNets -constraint_pins`
- `UnlockNets -constraint_pins -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `UnlockNets -constraint_pins -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `UnlockNets -constraint_pins -inst_or_protocol U1 U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `UnlockNets -nets {XP1_DDR_A0,XP1_DDR_A1}`
- `UnlockNets -bus XP1_DDR_A`
- `UnlockNets -inst_or_protocol U1`
- `UnlockNets -inst_or_protocol U1_U3`
- `UnlockNets -inst_or_protocol U1 -inst_or_protocol_group_or_bank B1`
- `UnlockNets -inst_or_protocol U2 -inst_or_protocol_group_or_bank group1`
- `UnlockNets -inst_or_protocol U1_U3 -inst_or_protocol_group_or_bank data_nibble`
- `UnlockNets -inst_or_protocol_group_or_bank U1_U3 data_nibble`

## Related Commands

- [LockNets](#)
- [GetAllLockedNetNames](#)

## UnPreservePairPins

Resets the preserve status of pair pins of the specified device instance.

### Return

bool

### Syntax

```
UnPreservePairPins deviceinstance
```

### Parameters

| Parameter          | Description                                                                                      | Type   | Optional |
|--------------------|--------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance of which the pair pins preserve status is to be reset. | string | false    |

## Examples

```
UnPreservePairPins U1
```

## Related Commands

[PreservePairPins](#)

# UnPreserveTrueDifferentialPins

Resets the preserve status of true differential pair pins of the specified device instance. This command is applicable to Actel devices.

## Return

bool

## Syntax

```
UnPreserveTrueDifferentialPins device_instance_name
```

## Parameters

| Parameter          | Description                                                                                        | Type   | Optional |
|--------------------|----------------------------------------------------------------------------------------------------|--------|----------|
| deviceInstanceName | Specifies the name of the device instance of which the true differential pair pins is to be reset. | string | false    |

## Examples

```
UnPreserveTrueDifferentialPins U2
```

## Related Commands

[PreserveTrueDifferentialPins](#)

# UpdateAllegroSchematics

Updates the pre-generated FPGA block schematics for the specified instance.

## Return

bool

## Syntax

```
UpdateAllegroSchematics instance_name
```

## Parameters

| Parameter     | Description                                                                      | Type   | Optional |
|---------------|----------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose FPGA block schematics is to be updated. | string | false    |

## Examples

```
UpdateAllegroSchematics U1
```

## Related Commands

- [GenerateOrCADSchematics](#)
- [GenerateOrCADSymbol](#)

## UpdateAssignedToPinsWithConnectedPins

This command updates the Assigned to Pins on the targeted interfaces/protocols/virtual interfaces of the specified device/connector with the connected pins. You can optionally restrict the scope of this operation to a specific interface/protocol/virtual interface or a specific logical group under them using the optional arguments.

## Return

bool

## Syntax

```
UpdateAssignedToPinsWithConnectedPins [-d device_or_connector_name] [-i interface_or_protocol_or_vi_name] [-g comma_separated_group_names]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                                                                                                                              | Type   | Optional |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -d        | Specifies the name of the device or connector for which the Assigned to Pins on the targeted interfaces/protocols/virtual interfaces have to be updated. If this argument is not specified, then the update operation is not restricted to the interfaces/protocols/virtual interfaces targeted to the specified device. | string | true     |
| -i        | Specifies the name of the interface, protocol or virtual interface on which the Assigned to Pins have to be updated. If this argument is not specified, then the update operation is not restricted to this interface/protocol/virtual interface.                                                                        | string | true     |
| -g        | Specifies the name of the group belonging to the specified interface, protocol or virtual interface on which the Assigned to Pins have to be updated. This argument is ignored if the -i argument is not specified. If this argument is not specified, then the update operation is not restricted to this group.        | string | true     |

## Examples

- `UpdateAssignedToPinsWithConnectedPins`
- `UpdateAssignedToPinsWithConnectedPins -d U2`
- `UpdateAssignedToPinsWithConnectedPins -d U2 -i XP1`
- `UpdateAssignedToPinsWithConnectedPins -d U2 -i XP1 -g Address_control`

## Related Commands

[ResetAssignedToPins](#)

## UpdateDeepNWideGroupName

Updates the existing Deep and Wide group name with the new specified group name.

## Return

bool

## Syntax

```
UpdateDeepNWideGroupName old_group_name new_group_name
```

## Parameters

| Parameter      | Description                                                                        | Type   | Optional |
|----------------|------------------------------------------------------------------------------------|--------|----------|
| old_group_name | Specifies the name of the existing Deep and Wide group.                            | string | false    |
| new_group_name | Specifies the new name that is to be updated for the existing deep and wide group. | string | false    |

## Examples

```
UpdateDeepNWideGroupName CommonGroup1 CG1
```

## Related Commands

- [CreateCommonGroup](#)
- [AddSecondary](#)
- [GetDeepNWideGroups](#)
- [RemoveDeepNWideGroup](#)
- [GetDeepNWideGroupInstanceList](#)

## UpdateDesignFromAllegro

Updates the current design by importing the Allegro board file.

## Return

bool

## Syntax

```
UpdateDesignFromAllegro -b allegro_board_file_path [-outline] [-placement] [-refdes]
```

## Parameters

| Parameter  | Description                                                                                                                                                                                                               | Type   | Optional |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -b         | Specifies the board file path that is to be used to update the current design.                                                                                                                                            | string | false    |
| -outline   | Specifies whether the PCB outline is to be updated. By default, outline is not updated.                                                                                                                                   | bool   | true     |
| -placement | Specifies whether the components placement is to be updated. By default, placement is not updated.                                                                                                                        | bool   | true     |
| -refdes    | Specifies whether the component and termination reference designators to be updated. By default, reference designators are not updated. Use this option to before generating second round of schematics in preserve mode. | bool   | true     |



## Examples

- `UpdateDesignFromAllegro -b ./project20.brd -outline`
- `UpdateDesignFromAllegro -b ./project20.brd -placement`
- `UpdateDesignFromAllegro -b ./project20.brd -refdes`
- `UpdateDesignFromAllegro -b ./project20.brd -outline -placement -refdes`

## Related Commands

[SetPCBOOutlineSettings](#)

# UpdateDeviceDataBase

Updates device instance database in order to execute the related data access tcl commands.

## Return

bool

## Syntax

```
UpdateDeviceDataBase device_instance_name
```

## Parameters

| Parameter      | Description                                | Type   | Optional |
|----------------|--------------------------------------------|--------|----------|
| deviceInstName | Specifies the name of the device instance. | string | false    |

## Examples

- `UpdateDeviceDataBase U1`
- `UpdateDeviceDataBase U3`

## Related Commands

- [InitDesignNetNameDatabase](#)
- [GetDeviceInstanceList](#)

# UpdateFPGAPortsFromCSV

Update FPGA Port names and Assigned to Pin for specified FPGA using CSV file

## Return

bool

## Syntax

```
UpdateFPGAPortsFromCSV -c csv_file_path -d delimiter -r reference_column -m column_mapping -a fpga_name [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                     | Type              | Optional |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -c        | Specifies the csv file path using which the FPGA ports need to be updated.                                                                                      | string            | no       |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), '(space) etc...                                 | string            | no       |
| -m        | Specifies the map between the Column name to column number.                                                                                                     | string_string_map | no       |
| -r        | Specifies the name of the reference column.                                                                                                                     | string            | no       |
| -a        | Specifies the name of the device instance for which the port names and assigned to pin is to be updated.                                                        | string            | no       |
| -i        | Specifies the row numbers in comma separated format, that are to be ignored during import from the CSV file. By default, this command does not ignore any rows. | string            | yes      |

## Examples

```
UpdateFPGAPortsFromCSV -c ./update_fpga_ports_csv.csv -d , -i 1 -r "Signal Name" -m {"Interface/Protocol Name" 1 "Signal Name" 2 "FPGA Port" 4 "Assigned to Pin" 5} -a U1
```

## Related Commands

- [ExportCSVFromFPGAPort](#)
- [MapPortNamestoPinNames](#)

## UpdateInstanceFootprint

Update instance pin location from given dra file. Please note that this command does not work for virtual interface and instances placed from component browser.

## Return

bool

## Syntax

```
UpdateInstanceFootprint instance_name dra_name
```

## Parameters

| Parameter     | Description                                  | Type   | Optional |
|---------------|----------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.          | string | false    |
| dra_name      | Specifies the dra name or complete dra path. | string | false    |

## Examples

- `UpdateInstanceFootprint U1 ff1152`
- `UpdateInstanceFootprint XP2 mlx-87705-001`

## Related Commands

- [GetInstanceFootprint](#)
- [GetDraPath](#)
- [IsDraExist](#)
- [GetDraDirectoriesPaths](#)
- [GetInstanceDRAAbsoluteFilePath](#)
- [ReportAllDRAFiles](#)
- [CheckDesignConsistency](#)

## UpdateInstanceLocation

Updates the placement location of the specified instance with the specified X and Y locations in the canvas.

### Return

bool

### Syntax

```
UpdateInstanceLocation instance_name newCenterXLoc newCenterYLoc rotation layer
```

### Parameters

| Parameter     | Description                                                                                           | Type   | Optional |
|---------------|-------------------------------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance.                                                                   | string | false    |
| newCenterXLoc | Specifies the center X coordinate location.                                                           | double | false    |
| newCenterYLoc | Specifies the center Y coordinate location.                                                           | double | false    |
| rotation      | Specifies the rotation degree with which the instance is to be rotated.                               | string | false    |
| layer         | Specifies the view to which the specified instance is to be flipped. Valid values are top and bottom. | string | false    |

### Examples

```
UpdateInstanceLocation XP1 5 5 50 bottom
```

## Related Commands

- [RotateInstance](#)
- [MoveInstance](#)
- [FlipInstance](#)
- [InitInstancePinLocation](#)

## UpdateInstancePartFromCSV

Updates the pin and group details of the specified instance by importing the CSV file.

## Return

bool

## Syntax

```
UpdateInstancePartFromCSV -l instance_or_connector_name -c csv_file_path -m column_mapping -r reference_column [-d delimiter] [-s instance_names] [-a save_as_file_path] [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                                     | Type              | Optional |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -l        | Specifies the name of the instance that is be updated.                                                                                                                          | string            | false    |
| -c        | Specifies the path of the csv file that is to used to update the instance pin properties.                                                                                       | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                                     | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                                     | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), '(space) etc...                                                 | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows.                  | string            | true     |
| -s        | Specifies the name of the other instances that are pointed to the specified rules file and is to updated. By default, this command considers to update only specified instance. | string            | true     |
| -a        | Specifies the path of the lrf file to be used to save the data of the instance part. This command updates the data by default to the same lrf that is pointed by instance.      | string            | true     |

## Examples

```
UpdateInstancePartFromCSV -c ./update_instance_part_csv.csv -d , -i 1 -r \"Pin Number\" -m {\"Pin Number\" 19 \"Target Pin Function\" 20} -l U3 -s U4 -a update_instance_part_csv_new_lrf.lrf
```

## Related Commands

- [GetInstanceNameList](#)
- [ExportCSVFromInstancePart](#)

## UpdateInstanceSymbolFromCSV

Updates the pin and symbol details of the specified instance by importing the CSV file.

## Return

bool

## Syntax

```
UpdateInstanceSymbolFromCSV -l instance_name -c csv_file_path -m column_mapping -r reference_column [-d delimiter] [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -l        | Specifies the name of the instance that is to be updated.                                                                                                      | string            | false    |
| -c        | Specifies the path of the csv file that is to be used to update the instance symbol pin properties.                                                            | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                    | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon), \t(tab), '(space) etc...                                | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

```
UpdateInstanceSymbolFromCSV -c ./update_instance_symbol_csv.csv -d , -i 1 -r \"Pin Number\" -m {\"Pin Number\" 1 \"Location\" 2} -l U3
```

## Related Commands

- [GetInstanceNameList](#)
- [ExportCSVSCchematicSymbolEditor](#)

## UpdateLayoutData

Updates the existing placement data file with the new data.

## Return

bool

## Syntax

```
UpdateLayoutData {instanceList}
```

## Parameters

| Parameter    | Description                                                                                                                                                                    | Type        | Optional |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------|
| instanceList | Specifies the list of instance names of which you want to update the layout data. For this command to run successfully, the specified instances must be present on the canvas. | string_list | false    |

## Examples

```
UpdateLayoutData {U1 XP1}
```

## Related Commands

- [GenerateLayoutData](#)
- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)

## UpdateNetGroup

Creates or updates netgroup in the design. Specify at least one command option.

### Return

bool

### Syntax

```
UpdateNetGroup -net_group_name [-nets {list_of_net_names}] [-bus bus_name] [-inst_or_protocol instance_or_protocol_name] [-inst_or_protocol_group_or_bank group_or_bank_name]
```

### Parameters

| Parameter                       | Description                                                                                                         | Type        | Optional |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------|-------------|----------|
| -net_group_name                 | Specifies net group name to be used for requested nets.                                                             | string      | false    |
| -nets                           | Creates or updates netgroup for the specified nets.                                                                 | string_list | true     |
| -bus                            | Creates or updates netgroup for nets belonging to specified bus name.                                               | string      | true     |
| -inst_or_protocol               | Creates or updates netgroup for the nets belonging to specified interface or protocol.                              | string      | true     |
| -inst_or_protocol_group_or_bank | Creates or updates netgroup for the nets belonging to specified group (or bank) of specified interface or protocol. | string      | true     |

### Examples

- UpdateNetGroup -net\_group\_name my\_new\_net\_group -nets [list XP1\_DDR\_A0 XP1\_DDR\_A1]
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -bus XP1\_DDR\_A
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol U1
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol U1\_U3
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol LA\*
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol U1 -inst\_or\_protocol\_group\_or\_bank B1
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol U2 -inst\_or\_protocol\_group\_or\_bank group1
- UpdateNetGroup -net\_group\_name my\_new\_net\_group -inst\_or\_protocol U1\_U3 -inst\_or\_protocol\_group\_or\_bank data\_nibble

### Related Commands

[CreateNewNetGroup](#)

## UpdateOrCADSchematics

Updates the existing FPGA block schematics for the specified instance.

### Return

bool

## Syntax

```
UpdateOrCADSchematics instance_name
```

## Parameters

| Parameter     | Description                                                                      | Type   | Optional |
|---------------|----------------------------------------------------------------------------------|--------|----------|
| instance_name | Specifies the name of the instance whose FPGA block schematics is to be updated. | string | false    |

## Examples

```
UpdateOrCADSchematics U1
```

## Related Commands

- [GenerateOrCADSchematics](#)
- [GenerateOrCADSymbol](#)

# UpdatePartDescription

Updates the description for the specified rules file.

## Return

bool

## Syntax

```
UpdatePartDescription rulesfilename partdescription
```

## Parameters

| Parameter       | Description                                                                      | Type   | Optional |
|-----------------|----------------------------------------------------------------------------------|--------|----------|
| rulesName       | Specifies the name of the rules file for which the description is to be updated. | string | false    |
| partDescription | Specifies the description of the part that is to be updated.                     | string | false    |

## Examples

```
UpdatePartDescription jtag \"JTAG interface for Xilinx devices\"
```

## Related Commands

[AddPartModel](#)

# UpdatePartFromCSV

Updates the existing part using the CSV file.

## Return

bool

## Syntax

```
UpdatePartFromCSV -l rules_file_path -c csv_file_path -m column_mapping -r reference_column_name [-d delimiter] [-i ignore_rows] [-g is_connector]
```

## Parameters

| Parameter | Description                                                                                                                                                                                                                                                | Type              | Optional |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -l        | Specifies the path of the rules file.                                                                                                                                                                                                                      | string            | false    |
| -c        | Specifies the path of the CSV file that is to be imported.                                                                                                                                                                                                 | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                                                                                                                | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                                                                                                                | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon).                                                                                                                                                     | string            | true     |
| -i        | Specifies the row numbers in comma separated format, that are to be ignored during import from the CSV file. By default, this command does not ignore any rows.                                                                                            | string            | true     |
| -g        | Specifies whether the specified rules file shall be of type connector or fixed pin interface. Use 'y' for generate connector rules file and 'n' for interface rules file. By default, this command considers specified rules file as interface rules file. | string            | true     |

## Examples

- UpdatePartFromCSV -l update\_part\_csv.lrf -c ./update\_part\_csv.csv -m {"Pin Number\" 19 \"Symbol Pin Name\" 28} -r \"Pin Number\" -d , -i 1
- UpdatePartFromCSV -l update\_part\_conn\_csv.lrf -c ./update\_part\_conn\_csv.csv -m {"Pin Number\" 19 \"Symbol Pin Name\" 28} -r \"Pin Number\" -d , -i 1 -g y

## Related Commands

[CreatePartFromCSV](#)

## UpdateProtocolFromCSV

Update existing protocol using CSV file.

## Return

bool

## Syntax

```
UpdateProtocolFromCSV -p protocol_name -a active_device_name -c csv_file_path -m column_mapping -r reference_column_name [-d delimiter] [-i ignore_rows]
```



## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -p        | Specifies the name of the protocol that need to be updated.                                                                                                    | string            | false    |
| -a        | Specifies the name of the device instance for which protocol definition need to be updated.                                                                    | string            | false    |
| -c        | Specifies the path and name of the file to be imported to update the protocol.                                                                                 | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                    | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon)                                                          | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

- UpdateProtocolFromCSV -p U1\_U2 -a U1 -c ./update\_protocol\_csv.csv -r "\"Port Name\"" -m {"\"Port Name\" 11 \"External Port\" 4 \"Target Pin Function\" 7}
- UpdateProtocolFromCSV -p U1\_U2 -a U1 -c ./update\_protocol\_csv.csv -r "\"Port Name\"" -m {"\"Port Name\" 11 \"External Port\" 4 \"Target Pin Function\" 7} -d | -r "\"1,2,3\""

## Related Commands

- [CreateProtocolFromCSV](#)
- [CreateProtocolFromLibraryModel](#)
- [CreateProtocolFromExistingProtocol](#)
- [CreateProtocolFromConstraintsPinoutfile](#)
- [ExportCSVFromProtocol](#)

## UpdateVIFromCSV

Update existing protocol using CSV file.

### Return

bool

### Syntax

```
UpdateVIFromCSV -v virtual_interface_name -c csv_file_path -m column_mapping -r reference_column [-d delimiter] [-i ignore_rows]
```

## Parameters

| Parameter | Description                                                                                                                                                    | Type              | Optional |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------|
| -v        | Specifies the name of the virtual interface instance that need to be updated.                                                                                  | string            | false    |
| -c        | Specifies the path of the csv file to be imported to update virtual interface.                                                                                 | string            | false    |
| -m        | Specifies the map between the column name to column number.                                                                                                    | string_string_map | false    |
| -r        | Specifies the name of the reference column.                                                                                                                    | string            | false    |
| -d        | Specifies the delimiter used in the CSV file. Valid values are ,(comma), (pipe),;(semicolon),:(colon).                                                         | string            | true     |
| -i        | Specifies the row numbers that are to be ignored during import from the CSV file in comma separated format. By default, this command does not ignore any rows. | string            | true     |

## Examples

```
UpdateVIFromCSV -c ./update_vi_csv.csv -d , -i 1 -r \"Port Name\" -m {\"Port Name\" 10 \"Onchip Termination\" 7} -v U2_VI26
```

## Related Commands

[CreateVIFromCSV](#)

# UpRevDesignDatabase

Upgrades the database of the FSP design created prior to 16.5 HF3 to the current release version. This command migrates the design database from multiple files to a single file (.fsp). Same can also be achieved in GUI by opening old database in FSP.

## Return

bool

## Syntax

```
UpRevDesignDatabase old_fsp_project_path new_fsp_project_path
```

## Parameters

| Parameter      | Description                                                                                                                                                            | Type   | Optional |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| oldProjectPath | Specifies the complete FSP database file path with .scp extension that is to be upgraded.                                                                              | string | false    |
| newProjectPath | Specifies the complete FSP database file path with .fsp extension where the migrated database is to be exported. New database file will be generated at this location. | string | false    |

## Examples

```
UpRevDesignDatabase D:/fsp_working/CUSTOMER_DESIGN/qualcomm/core.0912/core/core.scp D:/fsp_working/project_16_6_de_test/migrated_core.fsp
```

## Related Commands

[UpRevLibraryPartDatabase](#)

## UpRevLibraryPartDatabase

Upgrades the FSP library database created prior to 16.5 HF3 to the the current release. This command migrates the libraries models from multiple files to a single rules file database.

### Return

bool

### Syntax

```
UpRevLibraryPartDatabase old_fsp_project_library_path
```

### Parameters

| Parameter       | Description                         | Type   | Optional |
|-----------------|-------------------------------------|--------|----------|
| libsDefFilePath | Specifies the path to the libs.def. | string | false    |

### Examples

```
UpRevLibraryPartDatabase D:/fsp_working/CUSTOMER_DESIGN/qualcomm/core.0912/core/project_libs.def
```

### Related Commands

[UpRevDesignDatabase](#)

## Version

Returns the version number of the tool.

### Return

string

### Syntax

```
Version
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
Version
```

### Related Commands

[Version](#)

## ZoomFitAll

Zooms the board to display all the instances that are present on the board.

### Return

bool

### Syntax

```
ZoomFitAll
```

### Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

### Examples

```
ZoomFitAll
```

### Related Commands

[ZoomFitInstance](#)

## ZoomFitInstance

Zooms the board to display the complete specified instance.

### Return

bool

### Syntax

```
ZoomFitInstance instance_name
```

### Parameters

| Parameter                  | Description                                                   | Type   | Optional |
|----------------------------|---------------------------------------------------------------|--------|----------|
| <code>instance_name</code> | Specifies the name of the instance that you want to zoom fit. | string | false    |

### Examples

- `ZoomFitInstance U1`
- `ZoomFitInstance XP1`

### Related Commands

[ZoomFitAll](#)

---

# FlipAllDecapPortMapping

---

Flips decap port mapping by connecting port P1 to low power regulator and port P2 to high power regulator.

## Return

void

## Syntax

```
FlipAllDecapPortMapping
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
FlipAllDecapPortMapping
```

## Related Commands

- [ImportDecaps](#)
- [ExportDecaps](#)

# GenerateVirtualComponentPinMappingFile

---

Generates pin mapping file to relate virtual component of FSP (like virtual interface, hierarchical block instances) with actual component in Allegro. This command assumes net names and instance names are matching between FSP and Allegro board design.

## Return

bool

## Syntax

```
GenerateVirtualComponentPinMappingFile -board_file_name allegro_board_file_path [-
mapping_file output_mapping_file_path]
```

## Parameters

| Parameter            | Description                                                                                                                        | Type   | Optional |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------|--------|----------|
| -<br>board_file_name | Specifies (absolute or relative) path of board file.                                                                               | string | false    |
| -mapping_file        | Specifies path of mapping file to be updated. Incase file is not specified, FSP generates mapping file in local project directory. | string | true     |

## Examples

```
GenerateVirtualComponentPinMappingFile -board_file_name ../physical/design_pcie.brd -
mapping_file ./output/fsp_pin_mapping_file.txt
```

## Related Commands

[GetInstanceNameList](#)

---

# GetBusNamesOfGroup

---

Returns the names of all the different bus signals present in group.

## Return

string\_list

## Syntax

```
GetBusNamesOfGroup -i interface_instance_name_or_protocol_name -g group_name
```

## Parameters

| Parameter | Description                                                    | Type   | Optional |
|-----------|----------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or protocol name. | string | false    |
| -g        | Specifies the name of the group.                               | string | false    |

## Examples

- `GetBusNamesOfGroup -i U6 -g g1 -v CC`
- `SetTargetPinFunction -i U1_U2_U3 -g data_group`

## Related Commands

- [GetInstanceNameList](#)
- [ResetTargetPinFunction](#)

# GetMultiTargetDevices

Returns the list of the device instance names to which the specified interface group is targeted to.

## Return

string\_string\_list

## Syntax

```
GetMultiTargetDevices instance_name group_name
```

## Parameters

| Parameter     | Description                                | Type   | Optional |
|---------------|--------------------------------------------|--------|----------|
| instance_name | Specifies name of the instance.            | string | false    |
| group_name    | Specifies the name of the interface group. | string | false    |

## Examples

```
GetMultiTargetDevices XP1 Data_Output
```

## Related Commands

- [GetAllTargetDevice](#)
- [GetTargetInstanceListForDevice](#)
- [TargetDevice](#)
- [TargetToMultipleDevices](#)
- [GetGroupNameList](#)



---

# GetMultiTargettedGroupUseBanks

---

Returns a map of bank names to which the specified interface group is targeted.

## Return

string\_string\_map

## Syntax

```
GetMultiTargettedGroupUseBanks instance_name groupName
```

## Parameters

| Parameter     | Description                          | Type   | Optional |
|---------------|--------------------------------------|--------|----------|
| instance_name | Specifies the name of the interface. | string | false    |
| groupName     | Specifies the name of the group.     | string | false    |

## Examples

```
GetUseBanks XP2 Address_Control
```

## Related Commands

- [GetAllUseBanks](#)
- [GetUseBanksForProtocol](#)
- [GetDontUseBanks](#)
- [GetDontUseBanksForProtocol](#)
- [SetUseBanks](#)
- [SetUseBanksForProtocol](#)
- [SetUseBanksForMultiDevices](#)
- [SetDontUseBanks](#)
- [SetDontUseBanksForProtocol](#)

---

# GetPinNamesOfBus

---

Returns the names of all the bus signals present.

## Return

string\_list

## Syntax

```
GetPinNamesOfBus -i interface_instance_name_or_protocol_name -g group_name -b bus_name
```

## Parameters

| Parameter | Description                                                    | Type   | Optional |
|-----------|----------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or protocol name. | string | false    |
| -g        | Specifies the name of the group.                               | string | false    |
| -b        | Specifies the name of the bus.                                 | string | false    |

## Examples

- `GetPinNamesOfBus -i U6 -g g1 -b data`
- `GetPinNamesOfBus -i U1_U2_U3 -g data_group -b addr`

## Related Commands

- [GetInstanceNameList](#)
- [GetBusNamesOfGroup](#)

# GetTargetPinFunction

Returns the target pin function value of the corresponding interface instance pin or protocol signal.

## Return

string

## Syntax

```
GetTargetPinFunction -i interface_instance_name_or_protocol_name -p
pin_name_or_port_name -l protocol_target_device
```

## Parameters

| Parameter | Description                                                                    | Type   | Optional |
|-----------|--------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or protocol name.                 | string | false    |
| -p        | Specifies the signal name of the pin that you want to set target pin function. | string | false    |
| -l        | Specifies the device name.                                                     | string | true     |

## Examples

- `GetTargetPinFunction -i U6 -p A15`
- `GetTargetPinFunction -i U1_U2_U3 -p D0 -l U2`

## Related Commands

- [SetTargetPinFunction](#)
- [ResetTargetPinFunction](#)

---

# IsGroupTargettedToDevice

---

Specifies the group of the interface instance is targetted any device instance or not.

## Return

bool

## Syntax

```
IsGroupTargettedToDevice -i interface_instance_name -g group_name
```

## Parameters

| Parameter | Description                                                            | Type   | Optional |
|-----------|------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface for which the target is to be set. | string | false    |
| -g        | Specifies the name of the group.                                       | string | false    |

## Examples

```
IsGroupTargettedToDevice -i U2 -g Address_Control
```

## Related Commands

- [GetTargetDevice](#)
- [IsGroupTargettedToMultipleDevices](#)
- [TargetToMultipleDevices](#)

---

# IsGroupTargettedToMultipleDevices

---

Specifies the group of the interface instance is targetted to multiple devices or not.

## Return

bool

## Syntax

```
IsInterfacePrtocolGroup -i interface_instance_name -g group_name
```

## Parameters

| Parameter | Description                                                            | Type   | Optional |
|-----------|------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface for which the target is to be set. | string | false    |
| -g        | Specifies the name of the group.                                       | string | false    |

## Examples

```
IsGroupTargettedToMultipleDevices -i U2 -g Address_Control
```

## Related Commands

- [IsGroupTargettedToDevice](#)
- [GetTargetDevice](#)
- [IsInterfacePrtocolGroup](#)
- [TargetToMultipleDevices](#)

---

# IsInterfaceProtocolGroup

---

Specifies the group of the interface instance is part of interface protocol or not.

## Return

bool

## Syntax

```
IsInterfaceProtocolGroup -i interface_instance_name -g group_name
```

## Parameters

| Parameter | Description                                                            | Type   | Optional |
|-----------|------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface for which the target is to be set. | string | false    |
| -g        | Specifies the name of the group.                                       | string | false    |

## Examples

```
IsInterfaceProtocolGroup -i U2 -g Address_Control
```

## Related Commands

- [IsGroupTargettedToDevice](#)
- [GetTargetDevice](#)
- [IsGroupTargettedToMultipleDevices](#)
- [TargetToMultipleDevices](#)

---

# RenameProtocol

---

Rename protocol name in the design.

## Return

bool

## Syntax

```
RenameProtocol old_protocol_name new_protocol_name
```

## Parameters

| Parameter         | Description                                  | Type   | Optional |
|-------------------|----------------------------------------------|--------|----------|
| old_protocol_name | Specifies the existing name of the protocol. | string | false    |
| new_protocol_name | Specifies the new name of the protocol.      | string | false    |

## Examples

```
GetProtocolNames U1_U2 pcie_protocol
```



## Related Commands

- [GetProtocolNames](#)
- [ChangeProtocolOrder](#)
- [ExportCSVFromProtocol](#)
- [ExportProtocolDefinition](#)
- [ReOptimizeProtocol](#)
- [UpdateProtocolFromCSV](#)
- [SetAutoNetGroupProtocol](#)
- [GetInstanceNameList](#)
- [GetDeviceInstanceList](#)
- [GetInterfaceInstanceList](#)

---

# ReportCdsLibLibraryPaths

---

Reports library and its paths declared in cds.lib file.

## Return

bool

## Syntax

```
ReportCdsLibLibraryPaths
```

## Parameters

| Parameter | Description | Type | Optional |
|-----------|-------------|------|----------|
|-----------|-------------|------|----------|

## Examples

```
ReportCdsLibLibraryPaths
```

## Related Commands

- [ReportAllDRAFiles](#)
- [ReportAllMappingFiles](#)
- [ReportAllRulesFiles](#)
- [CheckDesignConsistency](#)

# ResetTargetPinFunction

Resets the specified interface instance pin target pin function value to part pin value.

## Return

bool

## Syntax

```
ResetTargetPinFunction interface_instance_name pin_number
```

## Parameters

| Parameter               | Description                                                                                       | Type   | Optional |
|-------------------------|---------------------------------------------------------------------------------------------------|--------|----------|
| interface_instance_name | Specifies the name of the interface instance.                                                     | string | false    |
| pin_number              | Specifies the pin number of the pin that you want to reset to part pin target pin function value. | string | false    |

## Examples

```
ResetTargetPinFunction U6 A15
```

## Related Commands

[GetInstanceNameList](#)

---

# SetTargetPinFunction

---

sets the specified target pin function value to the corresponding interface instance pin or protocol signal.

## Return

bool

## Syntax

```
SetTargetPinFunction -i interface_instance_name_or_protocol_name -p
pin_name_or_port_name -v target_pin_function_value -l protocol_target_device
```

## Parameters

| Parameter | Description                                                                    | Type   | Optional |
|-----------|--------------------------------------------------------------------------------|--------|----------|
| -i        | Specifies the name of the interface instance or protocol name.                 | string | false    |
| -p        | Specifies the signal name of the pin that you want to set target pin function. | string | false    |
| -v        | Specifies the value of the target pin function.                                | string | false    |
| -l        | Specifies the device name.                                                     | string | true     |

## Examples

- `SetTargetPinFunction -i U6 -p A15 -v CC`
- `SetTargetPinFunction -i U1_U2_U3 -p D0 -l U2 -v CC`

## Related Commands

- [GetInstanceNameList](#)
- [ResetTargetPinFunction](#)