GPStudio backend documentation

1.21

Generated by Doxygen 1.8.6

Fri Mar 3 2017 16:03:49

Contents

1 Main Page 2

1.1 model and backend . . . 2

1.2 user guide . . . 2

2 Module Index 2

2.1 Modules . . . 2

3 Hierarchical Index 2

3.1 Class Hierarchy . . . 2

4 Class Index 4

4.1 Class List . . . 4

5 Module Documentation 6

5.1 Base script model . . . 6

5.1.1 Detailed Description . . . 8

6 Class Documentation 8

6.1 Altera_quartus_toolchain Class Reference . . . 8

6.1.1 Member Function Documentation . . . 10

6.2 Attribute Class Reference . . . 12

6.2.1 Detailed Description . . . 13

6.2.2 Constructor & Destructor Documentation . . . 14

6.2.3 Member Function Documentation . . . 15

6.2.4 Member Data Documentation . . . 16

6.3 Block Class Reference . . . 16

6.3.1 Detailed Description . . . 20

6.3.2 Constructor & Destructor Documentation . . . 20

6.3.3 Member Function Documentation . . . 20

6.3.4 Member Data Documentation . . . 28

6.4 Block_generator Class Reference . . . 29

6.4.1 Constructor & Destructor Documentation . . . 30

6.4.2 Member Function Documentation . . . 30

6.4.3 Member Data Documentation . . . 31

6.5 Board Class Reference . . . 32

6.5.1 Detailed Description . . . 33

6.5.2 Constructor & Destructor Documentation . . . 33

6.5.3 Member Function Documentation . . . 34

6.5.4 Member Data Documentation . . . 35

6.6 Clock Class Reference . . . 36

6.6.1 Detailed Description . . . 38
6.6.2 Constructor & Destructor Documentation . . . 38
6.6.3 Member Function Documentation . . . 39
6.6.4 Member Data Documentation . . . 43
6.7 ClockDomain Class Reference . . . 44
6.7.1 Detailed Description . . . 45
6.7.2 Constructor & Destructor Documentation . . . 45
6.7.3 Member Function Documentation . . . 46
6.7.4 Member Data Documentation . . . 46
6.8 ClockInterconnect Class Reference . . . 47
6.8.1 Detailed Description . . . 50
6.8.2 Constructor & Destructor Documentation . . . 50
6.8.3 Member Function Documentation . . . 50
6.8.4 Member Data Documentation . . . 54
6.9 ComConnect Class Reference . . . 55
6.9.1 Detailed Description . . . 56
6.9.2 Constructor & Destructor Documentation . . . 56
6.9.3 Member Function Documentation . . . 56
6.9.4 Member Data Documentation . . . 57
6.10 ComDriver Class Reference . . . 57
6.10.1 Detailed Description . . . 59
6.10.2 Constructor & Destructor Documentation . . . 59
6.10.3 Member Function Documentation . . . 59
6.10.4 Member Data Documentation . . . 62
6.11 ComParam Class Reference . . . 63
6.11.1 Detailed Description . . . 63
6.11.2 Constructor & Destructor Documentation . . . 64
6.11.3 Member Function Documentation . . . 65
6.11.4 Member Data Documentation . . . 66
6.12 Component Class Reference . . . 66
6.12.1 Detailed Description . . . 71
6.12.2 Constructor & Destructor Documentation . . . 71
6.12.3 Member Function Documentation . . . 72
6.12.4 Member Data Documentation . . . 88
6.13 ComponentPart Class Reference . . . 90
6.13.1 Detailed Description . . . 91
6.13.2 Constructor & Destructor Documentation . . . 91
6.13.3 Member Function Documentation . . . 92
6.13.4 Member Data Documentation . . . 95
6.14 ComponentPartFlow Class Reference . . . 96

6.14.1 Detailed Description . . . 96
6.14.2 Constructor & Destructor Documentation . . . 97
6.14.3 Member Function Documentation . . . 97
6.14.4 Member Data Documentation . . . 98
6.15 ComponentPartProperty Class Reference . . . 98
6.15.1 Detailed Description . . . 99
6.15.2 Constructor & Destructor Documentation . . . 100
6.15.3 Member Function Documentation . . . 100
6.15.4 Member Data Documentation . . . 101
6.16 File Class Reference . . . 101
6.16.1 Detailed Description . . . 103
6.16.2 Constructor & Destructor Documentation . . . 103
6.16.3 Member Function Documentation . . . 103
6.16.4 Member Data Documentation . . . 104
6.17 Flow Class Reference . . . 105
6.17.1 Detailed Description . . . 106
6.17.2 Constructor & Destructor Documentation . . . 106
6.17.3 Member Function Documentation . . . 106
6.17.4 Member Data Documentation . . . 110
6.18 FlowConnect Class Reference . . . 110
6.18.1 Detailed Description . . . 111
6.18.2 Constructor & Destructor Documentation . . . 112
6.18.3 Member Function Documentation . . . 112
6.18.4 Member Data Documentation . . . 113
6.19 FlowInterconnect Class Reference . . . 113
6.19.1 Detailed Description . . . 116
6.19.2 Constructor & Destructor Documentation . . . 116
6.19.3 Member Function Documentation . . . 116
6.19.4 Member Data Documentation . . . 120
6.20 GPViewer Class Reference . . . 120
6.20.1 Detailed Description . . . 121
6.20.2 Constructor & Destructor Documentation . . . 122
6.20.3 Member Function Documentation . . . 123
6.20.4 Member Data Documentation . . . 124
6.21 HDL_toolchain Class Reference . . . 125
6.21.1 Member Function Documentation . . . 126
6.22 Info Class Reference . . . 127
6.22.1 Detailed Description . . . 128
6.22.2 Constructor & Destructor Documentation . . . 128
6.22.3 Member Function Documentation . . . 129

6.22.4 Member Data Documentation . . . 129
6.23 InterfaceBus Class Reference . . . 130
6.23.1 Detailed Description . . . 130
6.23.2 Constructor & Destructor Documentation . . . 130
6.23.3 Member Data Documentation . . . 131
6.24 IO Class Reference . . . 131
6.24.1 Detailed Description . . . 134
6.24.2 Constructor & Destructor Documentation . . . 134
6.24.3 Member Function Documentation . . . 135
6.24.4 Member Data Documentation . . . 136
6.25 IOCom Class Reference . . . 137
6.25.1 Detailed Description . . . 140
6.25.2 Constructor & Destructor Documentation . . . 140
6.25.3 Member Function Documentation . . . 140
6.25.4 Member Data Documentation . . . 141
6.26 Lib Class Reference . . . 141
6.26.1 Detailed Description . . . 143
6.26.2 Constructor & Destructor Documentation . . . 143
6.26.3 Member Function Documentation . . . 143
6.26.4 Member Data Documentation . . . 144
6.27 LibItem Class Reference . . . 145
6.27.1 Detailed Description . . . 145
6.27.2 Constructor & Destructor Documentation . . . 146
6.27.3 Member Data Documentation . . . 146
6.28 Module_param Class Reference . . . 146
6.28.1 Constructor & Destructor Documentation . . . 147
6.28.2 Member Data Documentation . . . 147
6.29 Node Class Reference . . . 147
6.29.1 Detailed Description . . . 148
6.29.2 Constructor & Destructor Documentation . . . 149
6.29.3 Member Function Documentation . . . 149
6.29.4 Member Data Documentation . . . 154
6.30 Param Class Reference . . . 155
6.30.1 Detailed Description . . . 157
6.30.2 Constructor & Destructor Documentation . . . 157
6.30.3 Member Function Documentation . . . 157
6.30.4 Member Data Documentation . . . 159
6.31 ParamBitfield Class Reference . . . 160
6.31.1 Detailed Description . . . 162
6.31.2 Constructor & Destructor Documentation . . . 162

6.31.3 Member Function Documentation . . . 162
6.31.4 Member Data Documentation . . . 164
6.32 ParamInterconnect Class Reference . . . 164
6.32.1 Detailed Description . . . 167
6.32.2 Constructor & Destructor Documentation . . . 167
6.32.3 Member Function Documentation . . . 167
6.32.4 Member Data Documentation . . . 168
6.33 Pin Class Reference . . . 168
6.33.1 Detailed Description . . . 169
6.33.2 Constructor & Destructor Documentation . . . 170
6.33.3 Member Function Documentation . . . 170
6.33.4 Member Data Documentation . . . 171
6.34 PLL Class Reference . . . 171
6.34.1 Detailed Description . . . 172
6.34.2 Constructor & Destructor Documentation . . . 173
6.34.3 Member Function Documentation . . . 173
6.34.4 Member Data Documentation . . . 173
6.35 Port Class Reference . . . 174
6.35.1 Detailed Description . . . 175
6.35.2 Constructor & Destructor Documentation . . . 175
6.35.3 Member Function Documentation . . . 175
6.35.4 Member Data Documentation . . . 177
6.36 Process Class Reference . . . 178
6.36.1 Detailed Description . . . 180
6.36.2 Constructor & Destructor Documentation . . . 180
6.36.3 Member Function Documentation . . . 181
6.37 Property Class Reference . . . 181
6.37.1 Detailed Description . . . 184
6.37.2 Constructor & Destructor Documentation . . . 184
6.37.3 Member Function Documentation . . . 185
6.37.4 Member Data Documentation . . . 191
6.38 PropertyEnum Class Reference . . . 192
6.38.1 Detailed Description . . . 193
6.38.2 Constructor & Destructor Documentation . . . 193
6.38.3 Member Function Documentation . . . 193
6.38.4 Member Data Documentation . . . 194
6.39 Reset Class Reference . . . 194
6.39.1 Detailed Description . . . 196
6.39.2 Constructor & Destructor Documentation . . . 196
6.39.3 Member Function Documentation . . . 196

6.39.4 Member Data Documentation . . . 197
6.40 Toolchain Class Reference . . . 197
6.40.1 Detailed Description . . . 200
6.40.2 Constructor & Destructor Documentation . . . 200
6.40.3 Member Function Documentation . . . 200
6.40.4 Member Data Documentation . . . 203
6.41 TreeConnect Class Reference . . . 203
6.41.1 Detailed Description . . . 204
6.41.2 Constructor & Destructor Documentation . . . 204
6.41.3 Member Function Documentation . . . 204
6.41.4 Member Data Documentation . . . 204
6.42 TreeItem Class Reference . . . 205
6.42.1 Constructor & Destructor Documentation . . . 205
6.42.2 Member Function Documentation . . . 205
6.42.3 Member Data Documentation . . . 206
6.43 VHDL_extractor Class Reference . . . 207
6.43.1 Constructor & Destructor Documentation . . . 207
6.43.2 Member Function Documentation . . . 208
6.43.3 Member Data Documentation . . . 210
6.44 VHDL_generator Class Reference . . . 211
6.44.1 Constructor & Destructor Documentation . . . 212
6.44.2 Member Function Documentation . . . 212
6.44.3 Member Data Documentation . . . 215
6.45 VHDLPart Class Reference . . . 216
6.45.1 Constructor & Destructor Documentation . . . 216
6.45.2 Member Data Documentation . . . 216
6.46 Viewer Class Reference . . . 217
6.46.1 Detailed Description . . . 218
6.46.2 Constructor & Destructor Documentation . . . 218
6.46.3 Member Function Documentation . . . 218
6.46.4 Member Data Documentation . . . 219
6.47 ViewerFlow Class Reference . . . 220
6.47.1 Detailed Description . . . 220
6.47.2 Constructor & Destructor Documentation . . . 221
6.47.3 Member Function Documentation . . . 221
6.47.4 Member Data Documentation . . . 222

Index **223**

1 Main Page

This document is the complete documentation of GPStudio backend. It composed of two parts :

- the model and backend develloper documentation
- the user guide documentation

1.1 model and backend

1.2 user guide

2 Module Index

2.1 Modules

Here is a list of all modules:

Base script model 6

3 Hierarchical Index

3.1 Class Hierarchy

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

Attribute 12

Block_generator 29

Board 32

Clock 36

ClockDomain 44

ComConnect 55

ComDriver 57

ComParam 63

Component 66

- **Block** 16
  - **ClockInterconnect** 47
  - **FlowInterconnect** 113
  - **IO** 131
    - **IOCom** 137
  - **ParamInterconnect** 164
  - **Process** 178

ComponentPart 90

ComponentPartFlow 96

ComponentPartProperty 98

File 101

Flow 105

FlowConnect 110

GPViewer 120

Info 127

InterfaceBus 130

Lib 141

LibItem 145

Module_param 146

Node 147

Param 155

ParamBitfield 160

Pin 168

PLL 171

Port 174

Property 181

PropertyEnum 192

Reset 194

Toolchain 197

HDL_toolchain 125

Altera_quartus_toolchain 8

TreeConnect 203

TreeItem 205

VHDL_extractor 207

VHDL_generator 211

VHDLPart 216

Viewer 217

ViewerFlow 220

4 Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Altera_quartus_toolchain 8

Attribute
Used to define special attributes for a toolchain 12

Block
Block is the base block definition for all blocks (IO, IOCom and process) 16

Block_generator 29

Board
Board is the class to load .dev support platform file 32

Clock
Used to define a clock source or a clock input in Board, Block or Component 36

ClockDomain
ClockDomain permits to save typical frequency of a clock domain 44

ClockInterconnect
ClockInterconnect is the generated block to manage all the clock in the node project 47

ComConnect
The ComConnect to define mapping between hardware interfaces and high level software tool 55

ComDriver
ComDriver is the specialised implementation of IO 57

ComParam
Gives all specifics parameters to software driver to etablish the connection and alk to the camera 63

Component
Component is the the definition of hardware components 66

ComponentPart
ComponentPart is the the graphical definition of hardware components 90

ComponentPartFlow
ComponentPart is the the graphical definition of flow (position, name) 96

ComponentPartProperty
ComponentPartProperty is the the graphical definition of a property (position, part) 98

File
Model class to have informations about file implementation in a Component 101

Flow
Model class to define flow hardware interface in a Component or Block 105

FlowConnect
Define flow connection between two flow interface of blocks, an output flow to an input flow interface 110

FlowInterconnect
FlowInterconnect is the generated block to manage all the flows in the node project 113

GPViewer
List of all the configurated viewer for gpviewer 120

HDL_toolchain 125

Info
Used to indicate information for IPs 127

InterfaceBus
Define a bus interface for PI 130

IO
IO is the specialised implementation of Block for sensors and communication blocks 131

IOCom
IOCom is the specialised implementation of IO 137

Lib
Lib is a container that store all the IPs available in library path 141

LibItem
LibItem is an item of the Lib container 145

Module_param 146

Node
Node is the base class container that store all the configuration of a node 147

Param
Param handle a constant parameter (generic for VHDL, param for verilog constant for C/C++) or register for hardware implementation 155

ParamBitfield
Bit field for param when param are registers 160

ParamInterconnect
ParamInterconnect is the generated block to manage all the parameter interfaces 164

Pin
Pin is the physical mapping between external port of an IO block and chip pins 168

PLL
PLL is a conveniant system to help CI PLL assignation and computation 171

Port
Port is external port definition for IO block 174

Process
Process is the specialised implementation of Block for processes 178

Property
Used to define high level properties 181

PropertyEnum
The PropertyEnum can be used to list the values that can take a property 192

Reset
Define a reset input or reset provider in a Component or Block 194

Toolchain
Toolchain class define a toolchain for building a project 197

TreeConnect
Define a connection between two flows 203

TreeItem 205

VHDL_extractor 207

VHDL_generator 211

VHDLPart 216

Viewer
Define a viewer input or viewer provider 217

ViewerFlow
Define a flow connection to a viewer 220

5 Module Documentation

5.1 Base script model

Classes

- class Attribute

  The Attribute class is used to define special attributes for a toolchain.
- class Block

  Block is the base block definition for all blocks (IO, IOCom and process).
- class Clock

  The Clock class is used to define a clock source or a clock input in Board, Block or Component.
- class ClockDomain

  ClockDomain permits to save typical frequency of a clock domain.
- class ComConnect

  The ComConnect to define mapping between hardware interfaces and high level software tool.
- class ComDriver

  ComDriver is the specialised implementation of IO.
- class ComParam

  The ComParam class gives all specifics parameters to software driver to etablish the connection and alk to the camera.
- class Component

  Component is the the definition of hardware components.
- class ComponentPart

  ComponentPart is the the graphical definition of hardware components.
- class ComponentPartFlow

  ComponentPart is the the graphical definition of flow (position, name)
- class ComponentPartProperty

  ComponentPartProperty is the the graphical definition of a property (position, part)
- class File

  Model class to have informations about file implementation in a Component.
- class Flow

  Model class to define flow hardware interface in a Component or Block.
- class FlowConnect

Define flow connection between two flow interface of blocks, an output flow to an input flow interface.

- class GPViewer

  List of all the configurated viewer for gpviewer.
- class Info

  The Info class is used to indicate information for IPs.
- class InterfaceBus

  Define a bus interface for PI.
- class IO

  IO is the specialised implementation of Block for sensors and communication blocks.
- class IOCom

  IOCom is the specialised implementation of IO.
- class Lib

  Lib is a container that store all the IPs available in library path.
- class LibItem

  LibItem is an item of the Lib container.
- class Node

  Node is the base class container that store all the configuration of a node.
- class Param

  Param handle a constant parameter (generic for VHDL, param for verilog constant for C/C++) or register for hardware implementation.
- class ParamBitfield

  Bit field for param when param are registers.
- class Pin

  Pin is the physical mapping between external port of an IO block and chip pins.
- class PLL

  PLL is a conveniant system to help CI PLL assignation and computation.
- class Port

  Port is external port definition for IO block.
- class Process

  Process is the specialised implementation of Block for processes.
- class Property

  The Property class is used to define high level properties.
- class PropertyEnum

  The PropertyEnum can be used to list the values that can take a property.
- class Reset

  The Reset class define a reset input or reset provider in a Component or Block.
- class Toolchain

  Toolchain class define a toolchain for building a project.
- class TreeConnect

  The TreeConnect class define a connection between two flows.
- class Viewer

  The Viewer class define a viewer input or viewer provider.
- class ViewerFlow

  The ViewerFlow class define a flow connection to a viewer.
- class ClockInterconnect

  ClockInterconnect is the generated block to manage all the clock in the node project.
- class FlowInterconnect

  FlowInterconnect is the generated block to manage all the flows in the node project.
- class ParamInterconnect

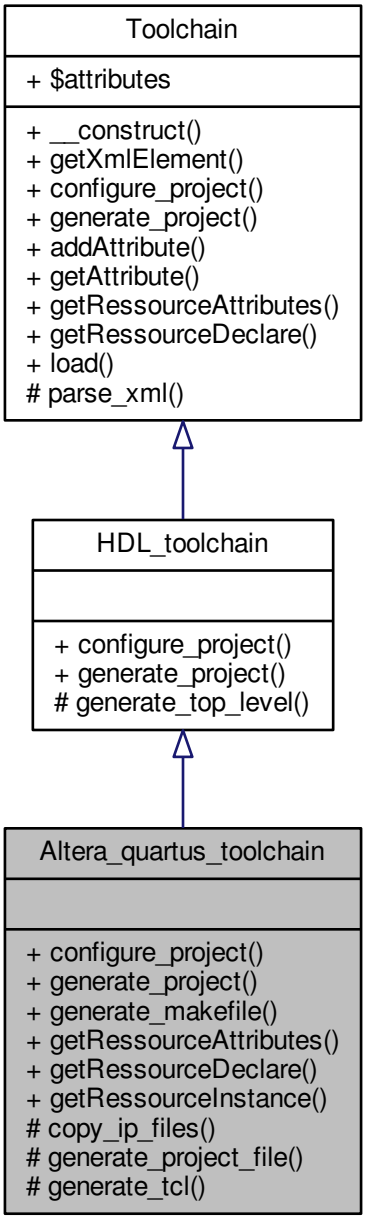
  ParamInterconnect is the generated block to manage all the parameter interfaces.

5.1.1 Detailed Description

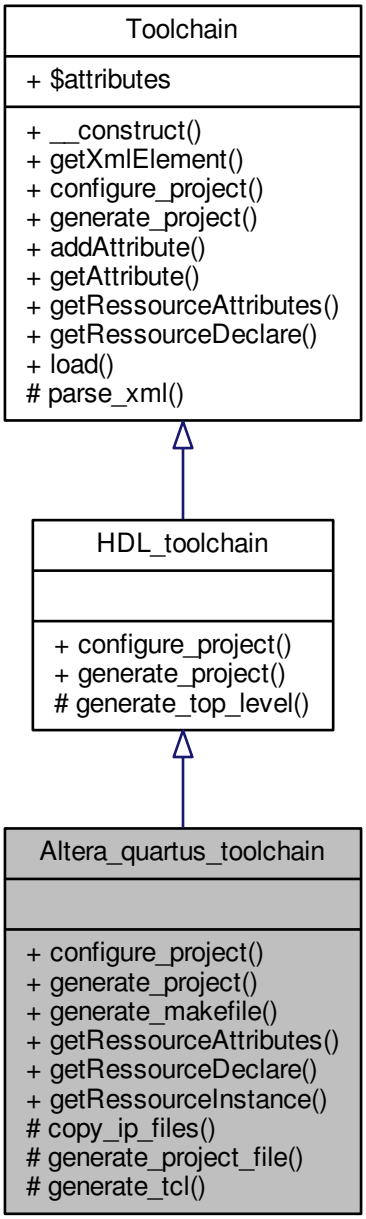
6 Class Documentation

6.1 Altera_quartus_toolchain Class Reference

Inheritance diagram for Altera_quartus_toolchain:



Collaboration diagram for Altera_quartus_toolchain:



Public Member Functions

- configure_project ($node)
- generate_project ($node, $path)
- generate_makefile ($node, $path)
- getRessourceAttributes ($type)
- getRessourceDeclare ($type, $params)
- getRessourceInstance ($type, $params)

Protected Member Functions

- copy_ip_files ($node, $path)
- generate_project_file ($node, $path)
- generate_tcl ($node, $path)

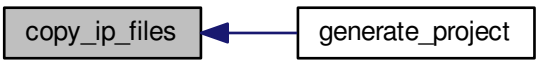
Additional Inherited Members

6.1.1 Member Function Documentation

6.1.1.1 configure_project (*$node*)

6.1.1.2 copy_ip_files (*$node, $path*) [protected]

Here is the caller graph for this function:

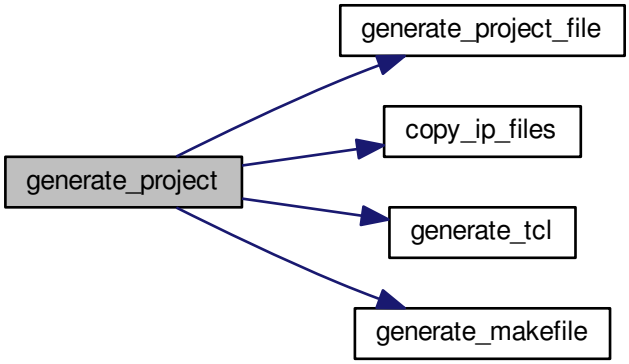


6.1.1.3 generate_makefile (*$node, $path*)

Here is the caller graph for this function:

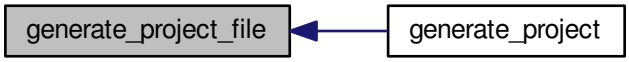
6.1.1.4 generate_project (*$node, $path*)

Here is the call graph for this function:



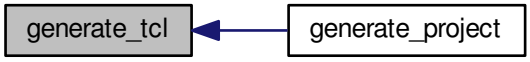
6.1.1.5 generate_project_file (*$node, $path*) [protected]

Here is the caller graph for this function:



6.1.1.6 generate_tcl (*$node, $path*) [protected]

Here is the caller graph for this function:



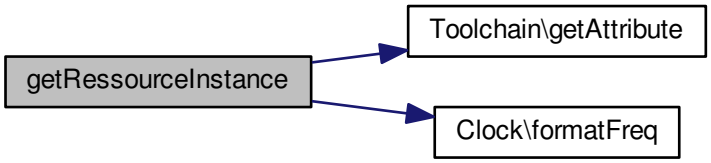
6.1.1.7 getRessourceAttributes (*$type*)

Here is the call graph for this function:

6.1.1.8 getRessourceDeclare (*$type, $params*)

6.1.1.9 getRessourceInstance (*$type, $params*)

Here is the call graph for this function:



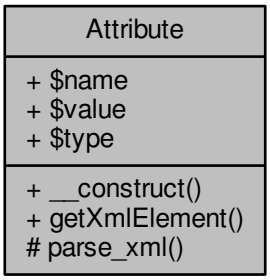
The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/altera_quartus/altera_quartus.php

6.2 Attribute Class Reference

The Attribute class is used to define special attributes for a toolchain.

Collaboration diagram for Attribute:



Public Member Functions

- __construct ($xml=null)

  Constructor of the class.
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the attribute.
- $value

  Value of the attibute.
- $type

  Type of attribute.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.2.1 Detailed Description

The Attribute class is used to define special attributes for a toolchain.

Attribute is contained into the Board class to define the specific attributes of the board and in Pin class to define each features of pin dedicated to the toolchain.

See Also

Toolchain Pin Board

6.2.2 Constructor & Destructor Documentation

6.2.2.1 __construct (*$xml* = `null`)

Constructor of the class.

Build an empty Attribute if $xml is empty, fill it with $xml if set

Parameters

SimpleXMLElement | null	*$xml*	XML element to parse if not null

Here is the call graph for this function:

6.2.3 Member Function Documentation

6.2.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.2.3.2 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.2.4 Member Data Documentation

6.2.4.1 string $name

Name of the attribute.

6.2.4.2 string $type

Type of attribute.

6.2.4.3 string $value

Value of the attibute.

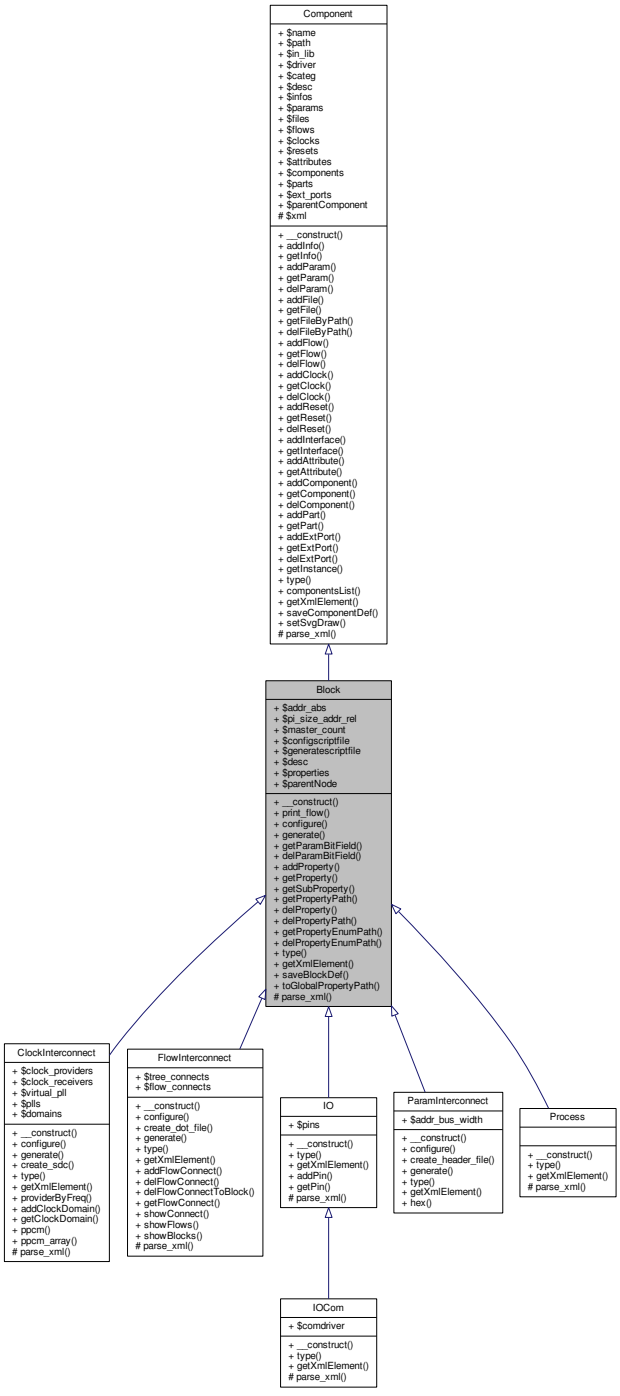
The documentation for this class was generated from the following file:

- model/attribute.php

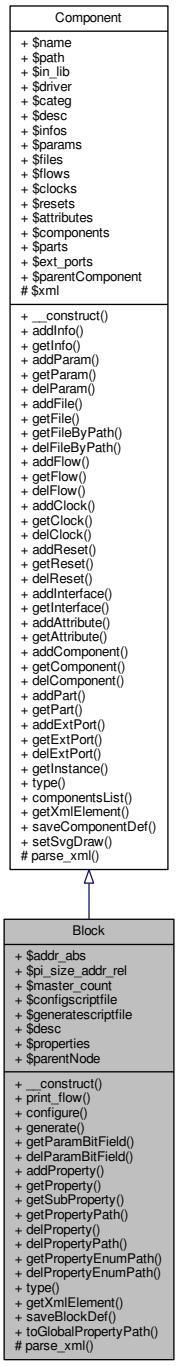
6.3 Block Class Reference

Block is the base block definition for all blocks (IO, IOCom and process).

Inheritance diagram for Block:



Collaboration diagram for Block:



Public Member Functions

- __construct ()

  Constructor of the class.
- print_flow ()

  Method to print a box corresponding to this block with all flow interface.
- configure ($node, $block)

Call the configure script if exist.

- generate ($node, $block, $path, $language)

  Call the generate script if exist.
- getParamBitField ($path, $casesens=true)

  return a reference to the bitfield with the path $path, if not found, return null
- delParamBitField ($path)

  return a reference to the bitfield with the path $path, if not found, return null
- addProperty ($property)

  Add a property to the block.
- getProperty ($name, $casesens=true)

  return a reference to the property with the name $name, if not found, return null
- getSubProperty ($name, $casesens=true)

  alias to getProperty($name, $casesens)
- getPropertyPath ($path, $casesens=true)

  return a reference to the property with the access path $path, if not found, return null
- delProperty ($name)

  delete a property from his name
- delPropertyPath ($path)

  delete a property from his path
- getPropertyEnumPath ($path, $casesens=true)

  return a reference to the property with the access path $path, if not found, return null
- delPropertyEnumPath ($path)

  delete a property enum from his path
- type ()

  Returns the type of the block as string, redefined by children.
- getXmlElement ($xml, $format)

  permits to output this instance
- saveBlockDef ($file)

  Helper function that save an '.io' or '.proc' file.
- toGlobalPropertyPath ()

  Redefines all the contained properties to the global context by adding the name of the block at the begining of all of properties map.

Public Attributes

- $addr_abs

  The absolute adress of the block on BI.
- $pi_size_addr_rel

  Size of relative adress bus.
- $master_count

  Number of master in the block on BI.
- $configscriptfile

  Specify the external file script to configure the block (optional)
- $generatescriptfile

  Specify the external file script to generate the block (optional)
- $desc

  Description of the block (optional)
- $properties

  Array of property class specify the high level properties.
- $parentNode

  Reference to the associated parent node.

Protected Member Functions

- parse_xml ($dummy=NULL, $dummy2=NULL)

  internal function to fill this instance from input xml structure

Additional Inherited Members

6.3.1 Detailed Description

Block is the base block definition for all blocks (IO, IOCom and process).

Block is a specialisation of component.

In addition of attribute of component, it contains bus interfaces for PI slave and master Block::$interfaces, addr and master count.

See Also

Component IO Process

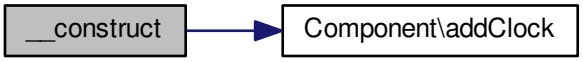
6.3.2 Constructor & Destructor Documentation

6.3.2.1 __construct ( )

Constructor of the class.

Build an empty Block with the default clock, 'clk_proc'.

Here is the call graph for this function:



6.3.3 Member Function Documentation

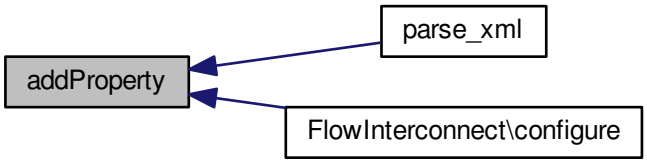
6.3.3.1 addProperty (*$property*)

Add a property to the block.

Parameters

Property	*$property*	property to add to the block

Here is the caller graph for this function:



6.3.3.2 configure (*$node, $block*)

Call the configure script if exist.

Parameters

Node	*$node*	node container
Block	*$block*	current block

See Also

$this->configscriptfile
configscriptfile

6.3.3.3 delParamBitField (*$path*)

return a reference to the bitfield with the path $path, if not found, return null

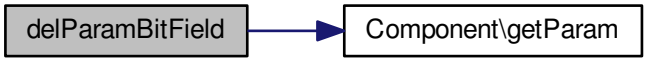
Parameters

string	*$path*	path of the parambitfield to search (param.parambitfield)

Returns

ParamBitfield found bitfield

Here is the call graph for this function:



6.3.3.4 delProperty (*$name*)

delete a property from his name

Parameters

string	*$name*	name of the property to delete

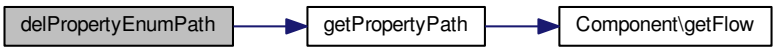
6.3.3.5 delPropertyEnumPath (*$path*)

delete a property enum from his path

Parameters

string	*$path*	path of the property to delete

Here is the call graph for this function:



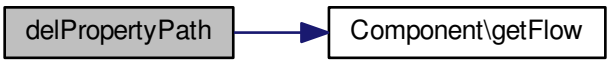
6.3.3.6 delPropertyPath (*$path*)

delete a property from his path

Parameters

string	*$path*	path of the property to delete

Here is the call graph for this function:



6.3.3.7 generate (*$node, $block, $path, $language*)

Call the generate script if exist.

Parameters

Node	*$node*	node container
Block	*$block*	current block
string	*$path*	path of the generated IP
string	*$language*	language of IP

See Also

$generatescriptfile

6.3.3.8 getParamBitField (*$path, $casesens* = `true`)

return a reference to the bitfield with the path $path, if not found, return null

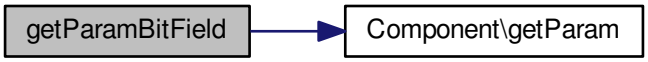
Parameters

string	*$path*	path of the parambitfield to search (param.parambitfield)
bool	*$casesens*	take care or not of the case of the name

Returns

ParamBitfield found bitfield

Here is the call graph for this function:



6.3.3.9 getProperty (*$name, $casesens* = `true`)

return a reference to the property with the name $name, if not found, return null

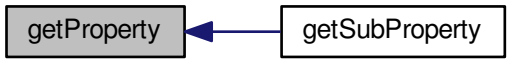
Parameters

string	*$name*	name of the property to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property

Here is the caller graph for this function:



6.3.3.10 getPropertyEnumPath (*$path, $casesens* = `true`)

return a reference to the property with the access path $path, if not found, return null

Parameters

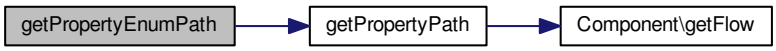
string	*$path*	path of the property to search, separed by . (dot)

bool	*$casesens*	take care or not of the case of the name

Returns

Property found property

Here is the call graph for this function:



6.3.3.11 getPropertyPath (*$path*, *$casesens* = `true`)

return a reference to the property with the access path $path, if not found, return null

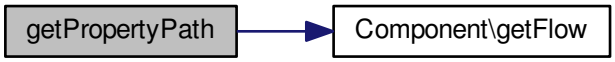
Parameters

string	*$path*	path of the property to search, separed by . (dot)
bool	*$casesens*	take care or not of the case of the name

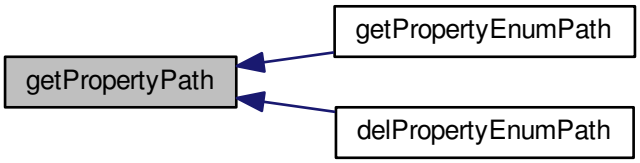
Returns

Property found property

Here is the call graph for this function:



Here is the caller graph for this function:



6.3.3.12 getSubProperty (*$name, $casesens* = `true`)

alias to getProperty($name, $casesens)

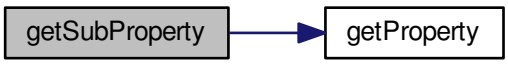
Parameters

string	*$name*	name of the property enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property

Here is the call graph for this function:



6.3.3.13 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

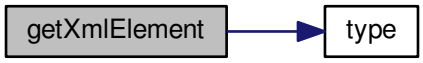
Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

Here is the call graph for this function:



Here is the caller graph for this function:

6.3.3.14 parse_xml (*$dummy* = NULL, *$dummy2* = NULL) [protected]

internal function to fill this instance from input xml structure

SimpleXMLElement $xml xml element to parse need to be specified in members before to call this function

Here is the call graph for this function:

6.3.3.15 print_flow ( )

Method to print a box corresponding to this block with all flow interface.

Ex :

| mt9 | out (8)

| |--------->

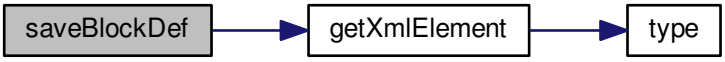
6.3.3.16 saveBlockDef (*$file*)

Helper function that save an '.io' or '.proc' file.

Parameters

string	*$file*	file name to save

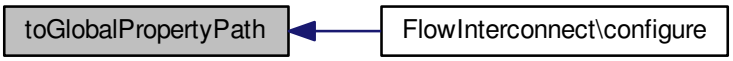
Here is the call graph for this function:



6.3.3.17 toGlobalPropertyPath ( )

Redefines all the contained properties to the global context by adding the name of the block at the begining of all of properties map.

Here is the caller graph for this function:



6.3.3.18 type ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

Here is the caller graph for this function:

6.3.4 Member Data Documentation

6.3.4.1 int $addr_abs

The absolute adress of the block on BI.

6.3.4.2 string $configscriptfile

Specify the external file script to configure the block (optional)

6.3.4.3 string $desc

Description of the block (optional)

6.3.4.4 string $generatescriptfile

Specify the external file script to generate the block (optional)

6.3.4.5 int $master_count

Number of master in the block on BI.

6.3.4.6 Node $parentNode

Reference to the associated parent node.

6.3.4.7 int $pi_size_addr_rel

Size of relative adress bus.

6.3.4.8 array Property $properties

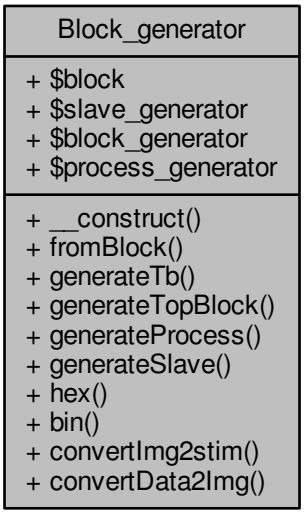
Array of property class specify the high level properties.

The documentation for this class was generated from the following file:

- model/block.php

6.4 Block_generator Class Reference

Collaboration diagram for Block_generator:



Public Member Functions

- __construct ($block=NULL)
- fromBlock ($block)
- generateTb ($path)
- generateTopBlock ($path)
- generateProcess ($path)
- generateSlave ($path)

Static Public Member Functions

- static hex ($value, $width)

- static bin ($value, $width)
- static convertImg2stim ($input_image, $output_file)
- static convertData2Img ($in, $out)

Public Attributes

- $block
- $slave_generator
- $block_generator
- $process_generator

6.4.1 Constructor & Destructor Documentation

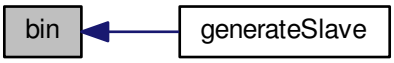
6.4.1.1 __construct (*$block* = `NULL`)

Here is the call graph for this function:

6.4.2 Member Function Documentation

6.4.2.1 static bin (*$value, $width*) `[static]`

Here is the caller graph for this function:



6.4.2.2 static convertData2Img (*$in, $out*) `[static]`

6.4.2.3 static convertImg2stim (*$input_image, $output_file*) `[static]`

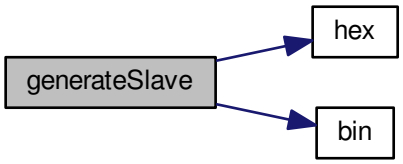
6.4.2.4 fromBlock (*$block*)

Here is the caller graph for this function:

6.4.2.5 generateProcess (*$path*)

6.4.2.6 generateSlave (*$path*)

Here is the call graph for this function:

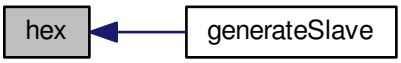


6.4.2.7 generateTb (*$path*)

6.4.2.8 generateTopBlock (*$path*)

6.4.2.9 static hex (*$value, $width*) [static]

Here is the caller graph for this function:



6.4.3 Member Data Documentation

6.4.3.1 $block

6.4.3.2 $block_generator

6.4.3.3 $process_generator

6.4.3.4 $slave_generator

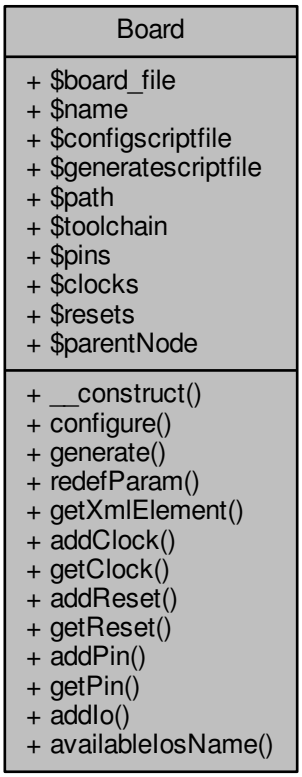
The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/block_generator.php

6.5 Board Class Reference

Board is the class to load .dev support platform file.

Collaboration diagram for Board:



Public Member Functions

- __construct ($board_element, $node)

  Constructor of the class.
- configure ($node)
- generate ($node, $path, $language)
- redefParam ($ios, $node)
- getXmlElement ($xml, $format)

permits to output this instance

- addClock ($clock)
- getClock ($name)
- addReset ($reset)
- getReset ($name)
- addPin ($pin)
- getPin ($name)
- addIo ($ioName)
- availableIosName ()

Public Attributes

- $board_file

  Complete path of the board definition.
- $name

  Name of the board.
- $configscriptfile

  Specify the external file script to configure the board (optional)
- $generatescriptfile

  Specify the external file script to generate the board files (optional)
- $path

  Path where the root of files and define of the board is putted.
- $toolchain

  Toolchain structure of board.
- $pins

  Array of pin mapping of the board.
- $clocks

  Array of input clocks of the board.
- $resets

  Array of input resets of the board.
- $parentNode

  Reference to the associated parent node.

6.5.1 Detailed Description

Board is the class to load .dev support platform file.

Board is the class to load .dev support platform file and manage IO support for the board. It also load the toolchain with attribute.

See Also

IO Pin Toolchain

6.5.2 Constructor & Destructor Documentation

6.5.2.1 __construct (*$board_element, $node*)

Constructor of the class.

Build an board from a board node elelement and a link to a node

Parameters

SimpleXMLElement | string	*$board_element*	XML element to parse if $board_element is an SimpleXMLElement object, name of the board else.
Node	*$node*	node associated to the board to parse it

6.5.3 Member Function Documentation

6.5.3.1 addClock (*$clock*)

Add a clock to the block

Parameters

Clock	*$clock*	clock to add to the block ∗

6.5.3.2 addIo (*$ioName*)

6.5.3.3 addPin (*$pin*)

Add a pin to the block

Parameters

Pin	*$pin*	pin to add to the block ∗

6.5.3.4 addReset (*$reset*)

Add a reset to the block

Parameters

Reset	*$reset*	reset to add to the block ∗

6.5.3.5 availableIosName ( )

6.5.3.6 configure (*$node*)

6.5.3.7 generate (*$node, $path, $language*)

6.5.3.8 getClock (*$name*)

return a reference to the clock with the name $name, if not found, return null

Parameters

string	*$name*	name of the clock to search

Returns

Clock found clock ∗

6.5.3.9 getPin (*$name*)

return a reference to the pin with the name $name, if not found, return false

Parameters

string	*$name*	name of the pin to search

Returns

Pin found pin ∗

6.5.3.10 getReset (*$name*)

return a reference to the reset with the name $name, if not found, return null

Parameters

string	*$name*	name of the reset to search

Returns

Reset found reset ∗

6.5.3.11 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.5.3.12 redefParam (*$ios, $node*)

Here is the call graph for this function:

6.5.4 Member Data Documentation

6.5.4.1 string $board_file

Complete path of the board definition.

6.5.4.2 array Clock $clocks

Array of input clocks of the board.

6.5.4.3 string $configscriptfile

Specify the external file script to configure the board (optional)

6.5.4.4 string $generatescriptfile

Specify the external file script to generate the board files (optional)

6.5.4.5 string $name

Name of the board.

6.5.4.6 Node $parentNode

Reference to the associated parent node.

6.5.4.7 string $path

Path where the root of files and define of the board is putted.

6.5.4.8 array Pin $pins

Array of pin mapping of the board.

6.5.4.9 array Reset $resets

Array of input resets of the board.

6.5.4.10 Toolchain $toolchain

Toolchain structure of board.

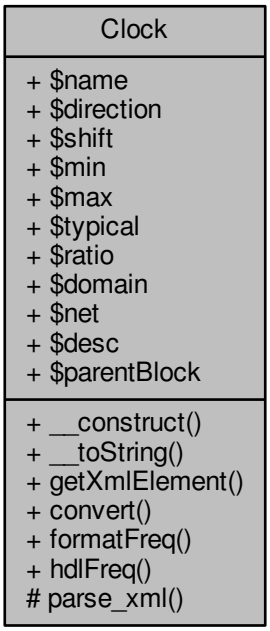
The documentation for this class was generated from the following file:

- model/board.php

6.6 Clock Class Reference

The Clock class is used to define a clock source or a clock input in Board, Block or Component.

Collaboration diagram for Clock:



Public Member Functions

- __construct ($xml=null)

  constructor of Clock
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Static Public Member Functions

- static convert ($string)

  frenquency from string
- static formatFreq ($freq)

  human readable frequency
- static hdlFreq ($freq)

  HDL readable frequency.

Public Attributes

- $name

  name of the clock
- $direction

  clock direction

- $shift
  clock shift phase
- $min
  minimum freq acceptance
- $max
  maximal freq acceptance
- $typical
  frequence of the clock
- $ratio
  ratio frequence in the clock domain
- $domain
  clock domain
- $net
  net to connect the clock
- $desc
  description
- $parentBlock
  parent block

Protected Member Functions

- parse_xml ($xml)
  internal function to fill this instance from input xml structure

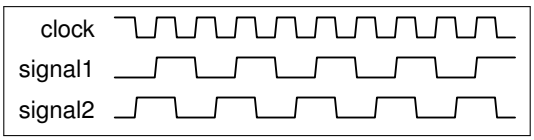
6.6.1 Detailed Description

The Clock class is used to define a clock source or a clock input in Board, Block or Component.

Clock instance is present in Board, Block or Component.

It's possible to define a clock with 3 ways :

- Clock::$typical frequency and eventualy a clock Clock::$shift given in degrees
- an interval of frequency given with Clock::$min and Clock::$max
- a clock Clock::$domain and a Clock::$ratio



If typical is set, the clock is completly constrained. In other case, the typical freq is computed by the ClockInterconnect::configure() method.

See Also

ClockInterconnect ClockDomain

6.6.2 Constructor & Destructor Documentation

6.6.2.1 __construct (*$xml* = `null`)

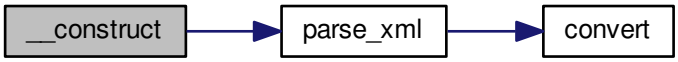
constructor of Clock

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.6.3 Member Function Documentation

6.6.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

Here is the call graph for this function:

6.6.3.2 static convert (*$string*) [static]

frenquency from string

Return a frequency as number in Hz from a string. Input format can be : "14.2M" or "18.7k" or "1500"

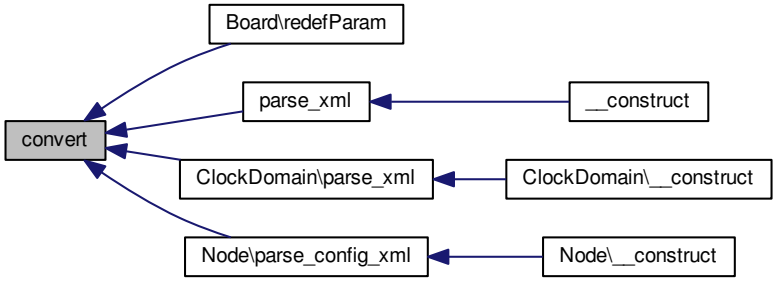
Parameters

string	*$string*	string to convert to frequency in Hz

Returns

int frequency in Hz

Here is the caller graph for this function:



6.6.3.3 static formatFreq (*$freq*) [static]

human readable frequency

Return formated string of a frequency for human reader. This is used for printable report or warning.

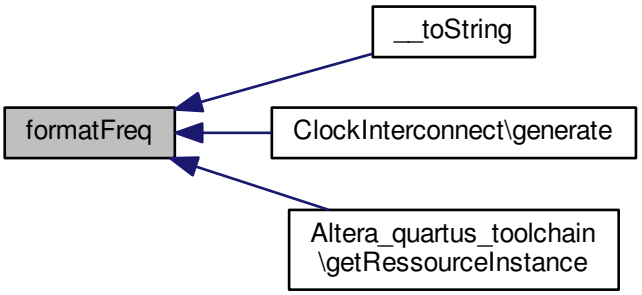
Parameters

int	*$freq*	frequency in Hz

Returns

string formated string

Here is the caller graph for this function:



6.6.3.4 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.6.3.5 static hdlFreq (*$freq*) [static]

HDL readable frequency.

Return formated string of a frequency for HDL compiler. This is used for naming convention in HDL output code.

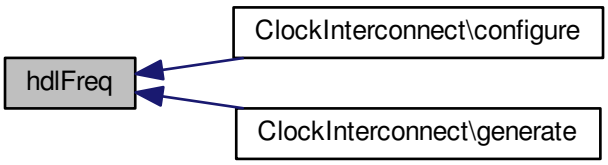
Parameters

int	*$freq*	frequency in Hz

Returns

string formated string

Here is the caller graph for this function:



6.6.3.6 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:

Here is the caller graph for this function:

6.6.4 Member Data Documentation

6.6.4.1 string $desc

description

Description of the clock (optional)

6.6.4.2 string $direction

clock direction

Specify if clock is in or out (value : "in" or "out", default "in")

6.6.4.3 string $domain

clock domain

Clocks in the same domain depend of the same clock source

6.6.4.4 float $max

maximal freq acceptance

Maximal value for this clock in Hz, could be written like this : "14.2M" or "18.7k" or "1500"

See Also

$min

6.6.4.5 float $min

minimum freq acceptance

Minimal value for this clock in Hz, could be written like this : "14.2M" or "18.7k" or "1500"

See Also

$max

6.6.4.6 string $name

name of the clock

This name should be unique in the block.

6.6.4.7 string $net

net to connect the clock

Physical net name to connect the clock. This value is computed by CI

See Also

ClockInterconnect

6.6.4.8 Block $parentBlock

parent block

Reference to the associated parent block

6.6.4.9 float $ratio

ratio frequence in the clock domain

ratio compared to the main clock domain. If domain not set, produce an error. Default value 1.

See Also

$domain

6.6.4.10 int $shift

clock shift phase

Phase shift of the clock given in degrees

6.6.4.11 float $typical

frequence of the clock

Typical value for this clock in Hz, could be written like this : "14.2M" or "18.7k" or "1500". If this field is set, min, max and ratio will be ignored.

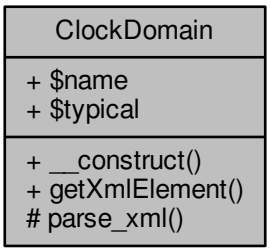
The documentation for this class was generated from the following file:

- model/clock.php

6.7 ClockDomain Class Reference

ClockDomain permits to save typical frequency of a clock domain.

Collaboration diagram for ClockDomain:



Public Member Functions

- __construct ($name=null, $typical=null)

constructor of ClockDomain

- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the clock.

- $typical

  Typical value for this clock in Hz, could be written like this : "14.2M" or "18.7k" or "1500".

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.7.1 Detailed Description

ClockDomain permits to save typical frequency of a clock domain.

ClockDomain permits to save typical frequency of a clock domain. Use it in addition to clock with shift or ratio. Clock defined with the same clockdomain ensure to be syncronized.

See Also

Clock ClockInterconnect

6.7.2 Constructor & Destructor Documentation

6.7.2.1 __construct (*$name* = `null`, *$typical* = `null`)

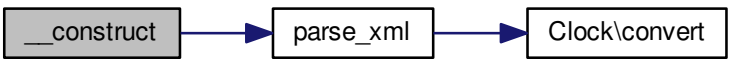
constructor of ClockDomain

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$name*	if it's different of null, call the xml parser to fill members, in case of $name is a string, init member with this value
int | null	*$typical*	

Here is the call graph for this function:



6.7.3 Member Function Documentation

6.7.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.7.3.2 parse_xml (*$xml*) [protected]

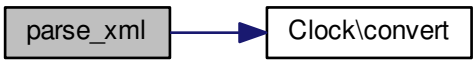
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.7.4 Member Data Documentation

6.7.4.1 string $name

Name of the clock.

6.7.4.2 float $typical

Typical value for this clock in Hz, could be written like this : "14.2M" or "18.7k" or "1500".

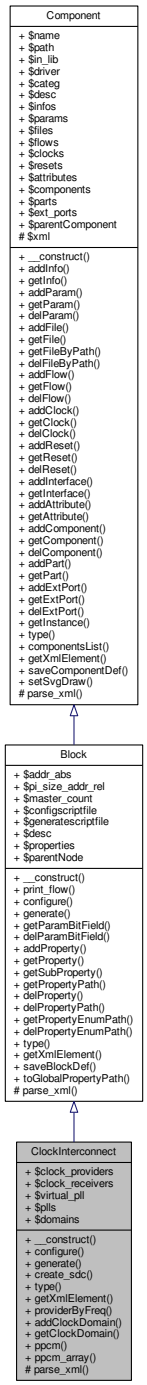
The documentation for this class was generated from the following file:

- model/clockdomain.php

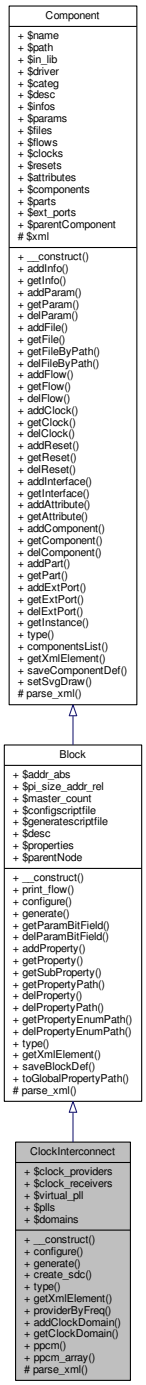
6.8 ClockInterconnect Class Reference

ClockInterconnect is the generated block to manage all the clock in the node project.

Inheritance diagram for ClockInterconnect:



Collaboration diagram for ClockInterconnect:



Public Member Functions

- __construct ()
- configure ($node, $block)
- generate ($node, $block, $path, $language)
- create_sdc ($path)
- type ()
- getXmlElement ($xml, $format)

- providerByFreq ($freq)
- addClockDomain ($domain)
- getClockDomain ($name)

Static Public Member Functions

- static ppcm ($nombre, $nombre2)
- static ppcm_array ($array, $a=0)

Public Attributes

- $clock_providers
- $clock_receivers
- $virtual_pll
- $plls
- $domains

Protected Member Functions

- parse_xml ($xml=NULL, $dummy2=NULL)

Additional Inherited Members

6.8.1 Detailed Description

ClockInterconnect is the generated block to manage all the clock in the node project.

All the clocks pass through this block. It gets all the base clock from board and IOs and provide all needed clock by using PLLs. Generated PLLs are located inside this block. If a clock is too slow, it generates a diviser based on a counter. The choice of clock association inside PLL are processed with an algorithm that minimise the number of PLLs.

See Also

Block Clock

6.8.2 Constructor & Destructor Documentation

6.8.2.1 __construct ( )

6.8.3 Member Function Documentation

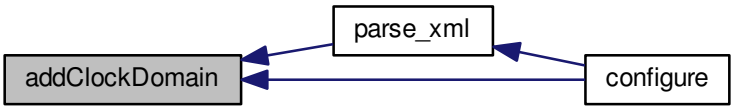
6.8.3.1 addClockDomain (*$domain*)

Add a clock domain to the block

Parameters

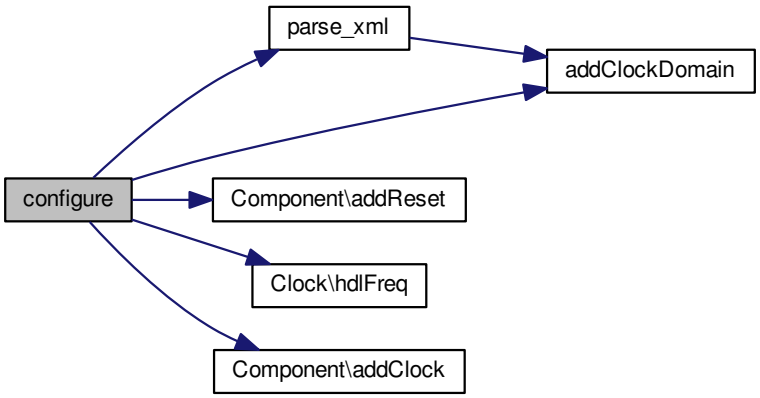
Clock-Domain	*$domain*	clock domain to add to the block *

Here is the caller graph for this function:



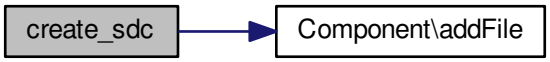
6.8.3.2 configure (*$node, $block*)

Here is the call graph for this function:



6.8.3.3 create_sdc (*$path*)

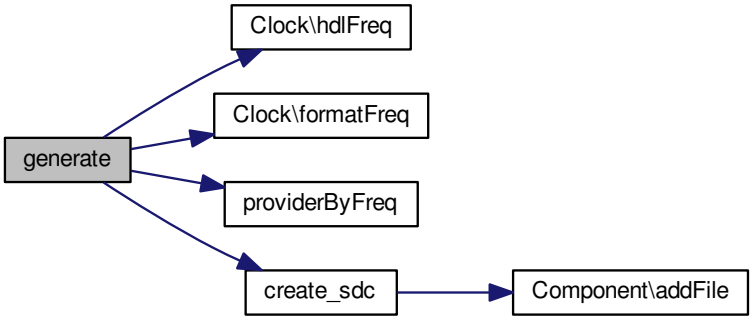
Here is the call graph for this function:



Here is the caller graph for this function:

6.8.3.4 generate (*$node, $block, $path, $language*)

Here is the call graph for this function:



6.8.3.5 getClockDomain (*$name*)

return a reference to the clock domain with the name $name, if not found, return null

Parameters

string	*$name*	name of the clock domain to search

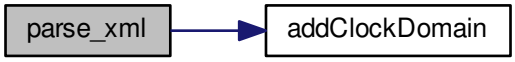
Returns

ClockDomain found clock domain *

6.8.3.6 getXmlElement (*$xml*, *$format*)

6.8.3.7 parse_xml (*$xml* = NULL, *$dummy2* = NULL) [protected]

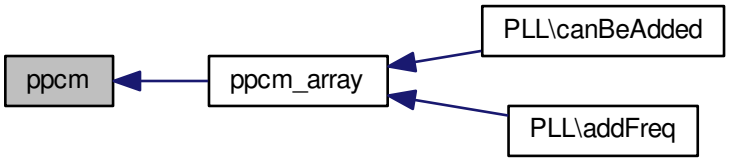
Here is the call graph for this function:



Here is the caller graph for this function:

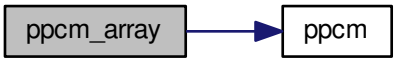
6.8.3.8 static ppcm (*$nombre*, *$nombre2*) [static]

Here is the caller graph for this function:

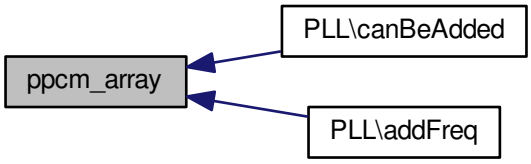


6.8.3.9 static ppcm_array (*$array, $a* = 0) [static]

Here is the call graph for this function:

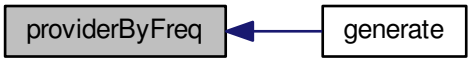


Here is the caller graph for this function:



6.8.3.10 providerByFreq (*$freq*)

Here is the caller graph for this function:



6.8.3.11 type ( )

6.8.4 Member Data Documentation

6.8.4.1 $clock_providers

6.8.4.2 $clock_receivers

6.8.4.3 $domains

6.8.4.4 $plls

6.8.4.5 $virtual_pll

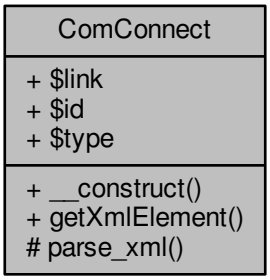
The documentation for this class was generated from the following file:

- system_interconnect/ci.php

6.9 ComConnect Class Reference

The ComConnect to define mapping between hardware interfaces and high level software tool.

Collaboration diagram for ComConnect:



Public Member Functions

- __construct ($xml=null)

  constructor of ComConnect
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $link

  Attribute name.
- $id

  Value of the id.
- $type

  Type of connection, could be "flow", "paramin" or "paramout".

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.9.1 Detailed Description

The ComConnect to define mapping between hardware interfaces and high level software tool.

It links the id of the software protocol to the name of the harware communication flow or PI interface

See Also

IOCom

6.9.2 Constructor & Destructor Documentation

6.9.2.1 __construct (*$xml* = `null`)

constructor of ComConnect

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.9.3 Member Function Documentation

6.9.3.1 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.9.3.2 parse_xml (*$xml*) `[protected]`

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.9.4 Member Data Documentation

6.9.4.1 string $id

Value of the id.

6.9.4.2 string $link

Attribute name.

6.9.4.3 string $type

Type of connection, could be "flow", "paramin" or "paramout".

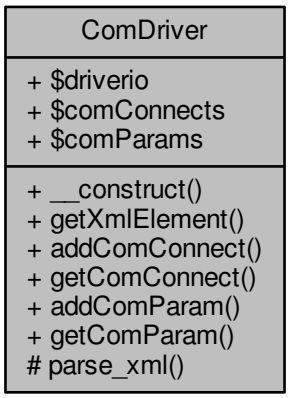
The documentation for this class was generated from the following file:

- model/comconnect.php

6.10 ComDriver Class Reference

ComDriver is the specialised implementation of IO.

Collaboration diagram for ComDriver:



Public Member Functions

- __construct ($xml=null)

  constructor of ComDriver
- getXmlElement ($xml, $format)

  permits to output this instance
- addComConnect ($comConnect)

  Add a comConnect to the comConnects.
- getComConnect ($link)

  return a reference to the comConnect with the link $link, if not found, return null
- addComParam ($comParam)

  Add a comParam to the comParams.
- getComParam ($name)

  return a reference to the comConnect with the link $link, if not found, return null

Public Attributes

- $driverio

  Name of the driver to use for etablish a communication with the board.
- $comConnects

  Array of ComConnect to give the equivalence table between hardware flow and software id flow.
- $comParams

  Array of ComParam to give all specific parameters.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.10.1 Detailed Description

ComDriver is the specialised implementation of IO.

Contains all parameters about the software driver to use and communication links and protocol declaration.

See Also

IOCom ComConnect ComParam

6.10.2 Constructor & Destructor Documentation

6.10.2.1 __construct (*$xml* = `null`)

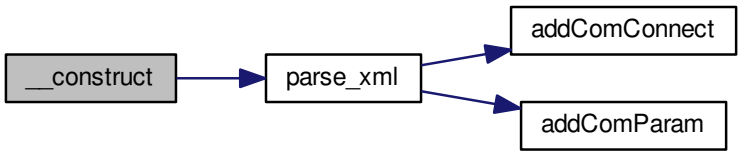
constructor of ComDriver

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.10.3 Member Function Documentation

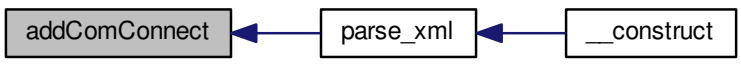
6.10.3.1 addComConnect (*$comConnect*)

Add a comConnect to the comConnects.

Parameters

ComConnect	*$comConnect*	comConnect to add to the comConnect

Here is the caller graph for this function:



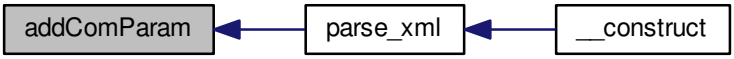
6.10.3.2 addComParam (*$comParam*)

Add a comParam to the comParams.

Parameters

ComParam	*$comParam*	comParam to add to the comParam

Here is the caller graph for this function:



6.10.3.3 getComConnect (*$link*)

return a reference to the comConnect with the link $link, if not found, return null

Parameters

string	*$link*	link of the comConnect to search

Returns

ComConnect found comConnect

6.10.3.4 getComParam (*$name*)

return a reference to the comConnect with the link $link, if not found, return null

Parameters

string	*$name*	link of the comConnect to search

Returns

ComConnect found comConnect

6.10.3.5 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

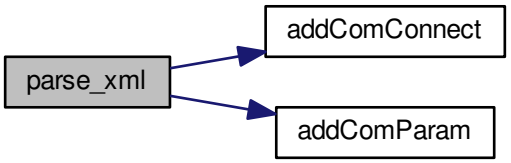
6.10.3.6 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Parameters

SimpleXMLElement	*$io_device_element*	element from io in lib
SimpleXMLElement	*$io_node_element*	element from the node

Here is the call graph for this function:



Here is the caller graph for this function:

6.10.4 Member Data Documentation

6.10.4.1 array **ComConnect** $comConnects

Array of ComConnect to give the equivalence table between hardware flow and software id flow.

See Also

ComConnect

6.10.4.2 array **ComParam** $comParams

Array of ComParam to give all specific parameters.

See Also

ComParam

6.10.4.3 string $driverio

Name of the driver to use for etablish a communication with the board.

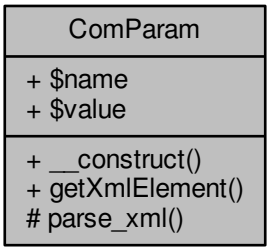
The documentation for this class was generated from the following file:

- model/comdriver.php

6.11 ComParam Class Reference

The ComParam class gives all specifics parameters to software driver to etablish the connection and alk to the camera.

Collaboration diagram for ComParam:



Public Member Functions

- __construct ($xml=null)

  constructor of ComParam
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Attribute name.
- $value

  Value of the parameter.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.11.1 Detailed Description

The ComParam class gives all specifics parameters to software driver to etablish the connection and alk to the camera.

See Also

IODriver

6.11.2 Constructor & Destructor Documentation

6.11.2.1 __construct (*$xml* = `null`)

constructor of ComParam

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.11.3 Member Function Documentation

6.11.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.11.3.2 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.11.4 Member Data Documentation

6.11.4.1 string $name

Attribute name.

6.11.4.2 string $value

Value of the parameter.

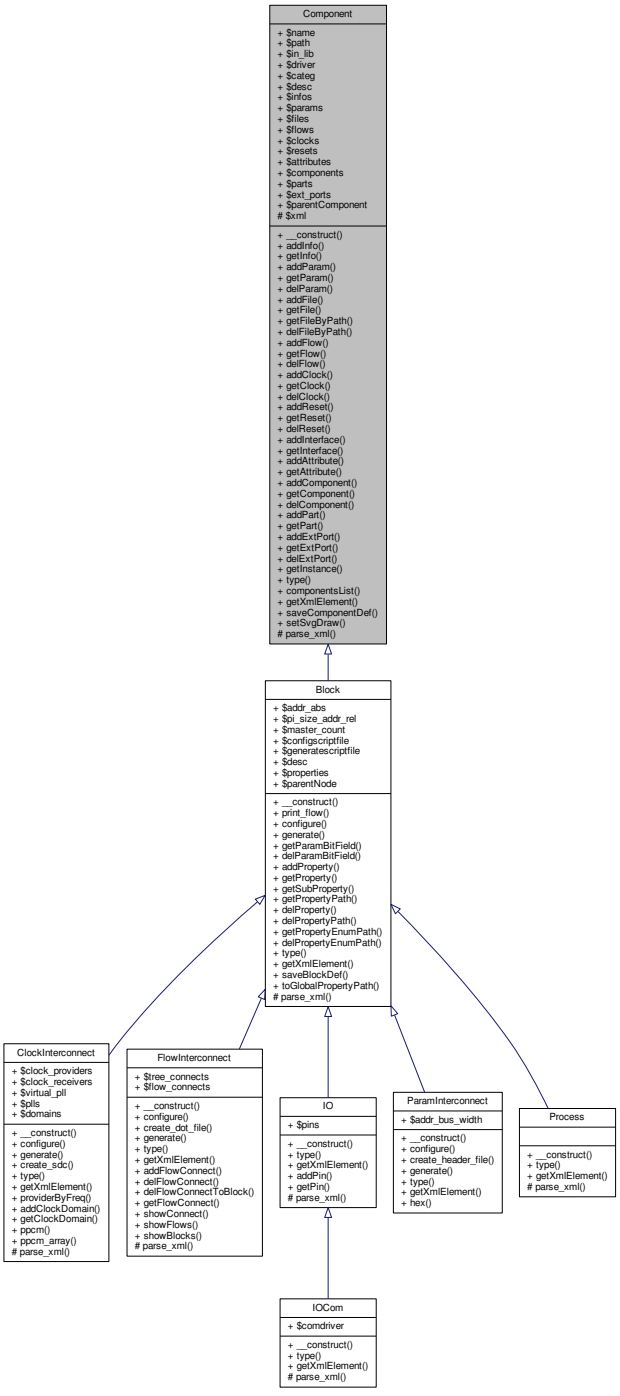
The documentation for this class was generated from the following file:

- model/comparam.php

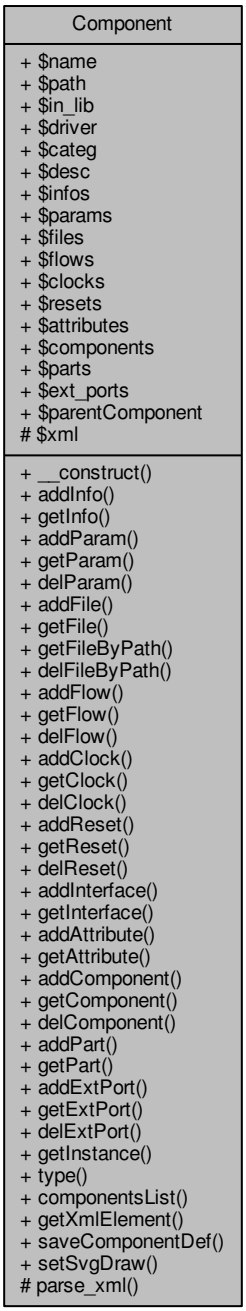
6.12 Component Class Reference

Component is the the definition of hardware components.

Inheritance diagram for Component:



Collaboration diagram for Component:



Public Member Functions

- __construct ($component=NULL)

  constructor of Component
- addInfo ($info)

  Add an info to the toolchain.
- getInfo ($name, $casesens=true)

return a reference to the component with the name $name, if not found, return null

- addParam ($param)

  Add a parameter to the component.
- getParam ($name, $casesens=true)

  return a reference to the parameter with the name $name, if not found, return null
- delParam ($name)

  delete a param from his name
- addFile ($file)

  Add a file to the component.
- getFile ($name)

  return a reference to the file with the name $name, if not found, return null
- getFileByPath ($path)

  return a reference to the file with the path $path, if not found, return null
- delFileByPath ($path)

  delete a file from his path
- addFlow ($flow)

  Add a flow to the component.
- getFlow ($name, $casesens=true)

  return a reference to the flow with the name $name, if not found, return null
- delFlow ($name)

  delete a flow from his name
- addClock ($clock)

  Add a clock to the component.
- getClock ($name, $casesens=true)

  return a reference to the clock with the name $name, if not found, return null
- delClock ($name)

  delete a clock from his name
- addReset ($reset)

  Add a reset to the component.
- getReset ($name, $casesens=true)

  return a reference to the reset with the name $name, if not found, return null
- delReset ($name)

  delete a reset from his name
- addInterface ($interface)

  Add an InterfaceBus to the block.
- getInterface ($name, $casesens=true)

  return a reference to the interface with the name $name, if not found, return null
- addAttribute ($attribute)

  Add an attribute to the toolchain.
- getAttribute ($name, $casesens=true)

  return a reference to the attribute with the name $name, if not found, return null
- addComponent ($component)

  Add a sub component.
- getComponent ($name, $casesens=true)

  return a reference to the component with the name $name, if not found, return null
- delComponent ($driver)

  delete a component support from his driver name
- addPart ($part)

  Add a gui part.
- getPart ($name, $casesens=true)

  return a reference to the part with the name $name, if not found, return null

- addExtPort ($extPort)

  Add an external port to the block.
- getExtPort ($name, $casesens=true)

  return a reference to the external port with the name $name, if not found, return null
- delExtPort ($name)

  delete an external port from his name
- getInstance ($name, $casesens=true)

  return a reference to the instance with the name $name, if not found, return null
- type ()

  Returns the type of the block as string, redefined by children.
- componentsList ()

  Returns the type of the block as string, redefined by children.
- getXmlElement ($xml, $format)

  permits to output this instance
- saveComponentDef ($file)

  Helper function that save an '.comp' file.
- setSvgDraw ($svgXml, $partName="main")

Public Attributes

- $name

  Name of the component.
- $path

  Path where the root of files and define of the component is putted.
- $in_lib

  Specify if the component is defined in the library or not.
- $driver

  Specify the name of the driver for the component.
- $categ

  Specify the categorie of the component eg : communication, imagesensor, descriptor...
- $desc

  Description of the flow (optional)
- $infos

  Array of information on component.
- $params

  Array of parameters class (can be Generic or dynamics parameter on BI)
- $files

  Array of files whith define the implementation of the component.
- $flows

  Array of flows in the component can be input flow or output.
- $clocks

  Array of clocks to drive the component.
- $resets

  Array of resets, can be different type of resets.
- $attributes

  Array of attributes.
- $components

  Array of children components.
- $parts

  Array of graphical part.
- $ext_ports

Array of port abble to comunicate with the output.

- $parentComponent

  Parent components, null if the component does not have a parent.

Protected Member Functions

- parse_xml ($dummy=NULL, $dummy2=NULL)

  internal function to fill this instance from input xml structure

Protected Attributes

- $xml

6.12.1 Detailed Description

Component is the the definition of hardware components.

Component is the the definition of hardware components. It could be used in a block to indicate the inclusion of the block or in the component library to define them.

It needs to be specialised, it only contains the list of :

- implementation files (vhdl, verilog, C, C++, ...), documentation files Block::$files
- hardware parameters (generic for VHDL, param for verilog constant for C/C++) or register for hardware implementation Block::$params
- properties for high level software Block::$properties
- flows interface input and output Block::$flows
- clocks inputs (all blocks) and clocks generator (IO blocks only) Block::$clocks
- reset inputs (all blocks) and reset generators (IO blocks only) Block::$resets
- attributes for special attributes compilation toolchain Block::$attributes

A component could be included in a block or as an extension but could not be instantiated in a node.

See Also

Block

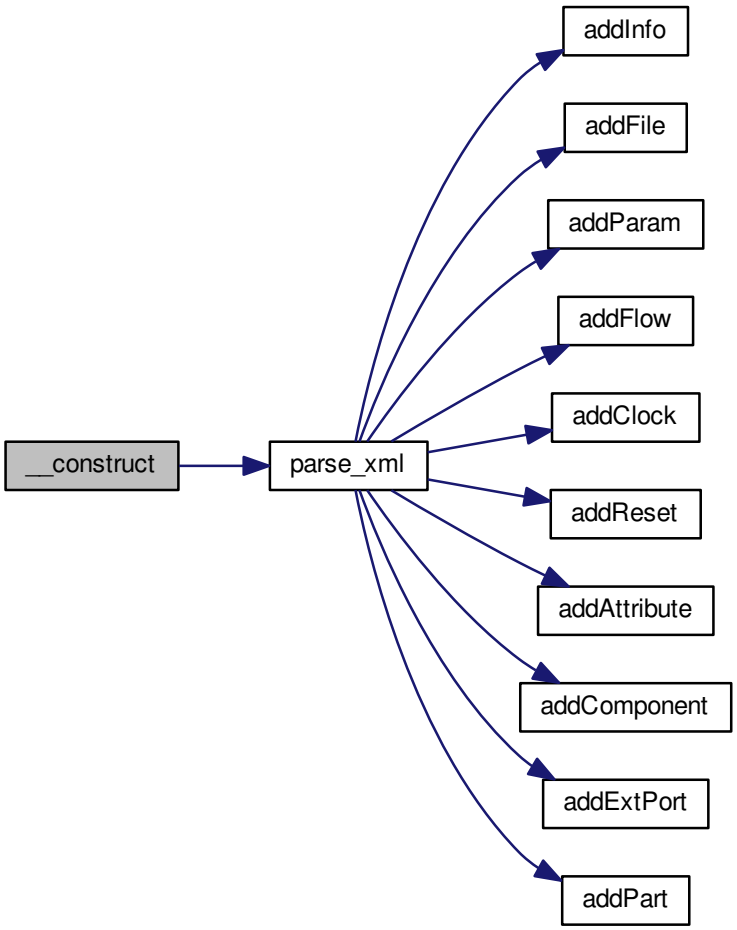
6.12.2 Constructor & Destructor Documentation

6.12.2.1 __construct (*$component* = NULL)

constructor of Component

Initialise all the internal members

Here is the call graph for this function:



6.12.3 Member Function Documentation

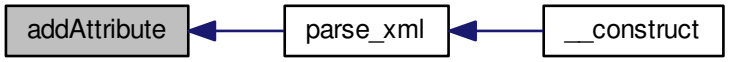
6.12.3.1 addAttribute (*$attribute*)

Add an attribute to the toolchain.

Parameters

Attribute	*$attribute*	attribute to add to the block

Here is the caller graph for this function:



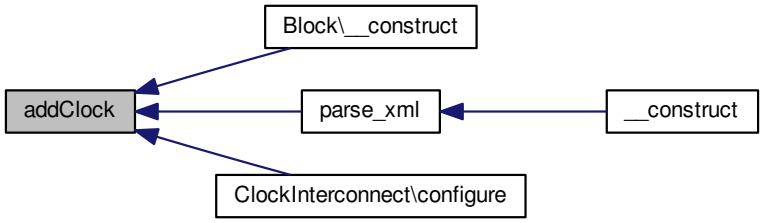
6.12.3.2 addClock (*$clock*)

Add a clock to the component.

Parameters

Clock	*$clock*	clock to add to the component

Here is the caller graph for this function:



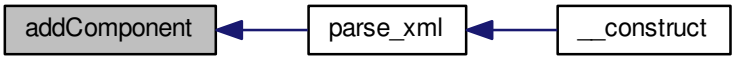
6.12.3.3 addComponent (*$component*)

Add a sub component.

Parameters

Component	*$component*	component to add to the block

Here is the caller graph for this function:



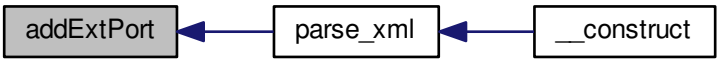
6.12.3.4 addExtPort (*$extPort*)

Add an external port to the block.

Parameters

Port	*$extPort*	port to add to the block

Here is the caller graph for this function:



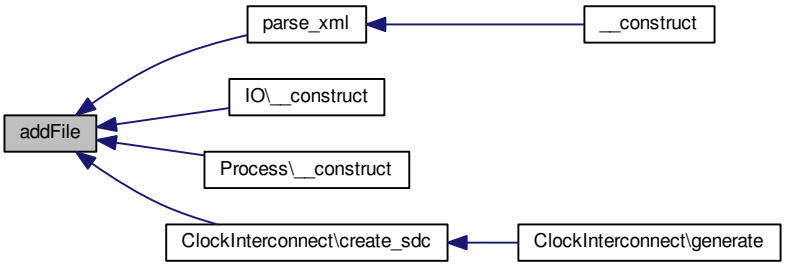
6.12.3.5 addFile (*$file*)

Add a file to the component.

Parameters

File	*$file*	file to add to the component

Here is the caller graph for this function:



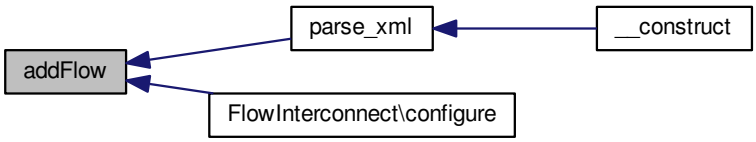
6.12.3.6 addFlow (*$flow*)

Add a flow to the component.

Parameters

Flow	*$flow*	flow to add to the component

Here is the caller graph for this function:



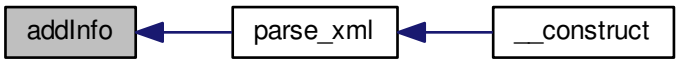
6.12.3.7 addInfo (*$info*)

Add an info to the toolchain.

Parameters

Info	*$info*	info to add to the component

Here is the caller graph for this function:



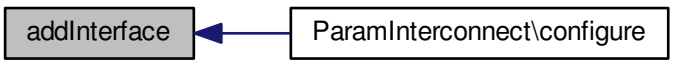
6.12.3.8 addInterface (*$interface*)

Add an InterfaceBus to the block.

Parameters

Interface-Bus	*$interface*	interface to add to the block

Here is the caller graph for this function:



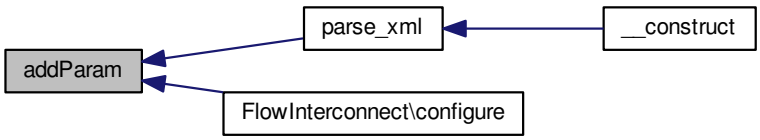
6.12.3.9 addParam (*$param*)

Add a parameter to the component.

Parameters

Param	*$param*	parameter to add to the component

Here is the caller graph for this function:



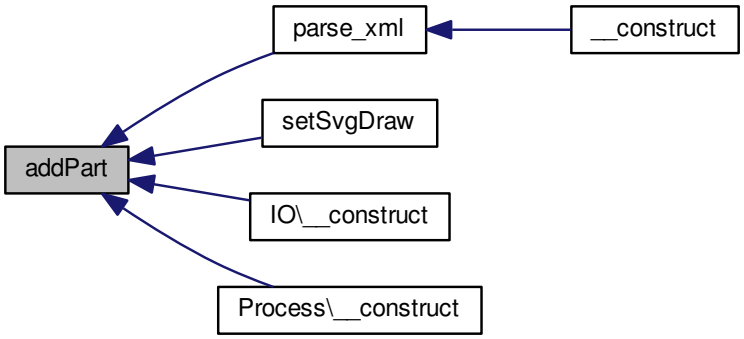
6.12.3.10 addPart (*$part*)

Add a gui part.

Parameters

Component-Part	*$part*	component to add to the block

Here is the caller graph for this function:



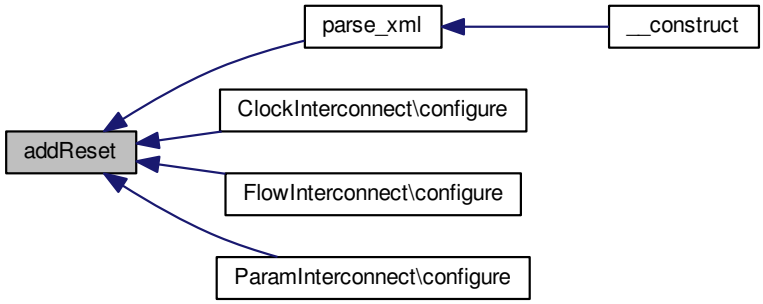
6.12.3.11 addReset (*$reset*)

Add a reset to the component.

Parameters

Reset	*$reset*	reset to add to the component

Here is the caller graph for this function:



6.12.3.12 componentsList ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

6.12.3.13 delClock (*$name*)

delete a clock from his name

Parameters

string	*$name*	name of the clock to delete

6.12.3.14 delComponent (*$driver*)

delete a component support from his driver name

Parameters

string	*$driver*	name of the reset to delete

6.12.3.15 delExtPort (*$name*)

delete an external port from his name

Parameters

string	*$name*	name of the external port to delete

6.12.3.16 delFileByPath (*$path*)

delete a file from his path

Parameters

string	*$path*	path of the file to delete

6.12.3.17 delFlow (*$name*)

delete a flow from his name

Parameters

string	*$name*	name of the flow to delete

6.12.3.18 delParam (*$name*)

delete a param from his name

Parameters

string	*$name*	name of the param to delete

6.12.3.19 delReset (*$name*)

delete a reset from his name

Parameters

string	*$name*	name of the reset to delete

6.12.3.20 getAttribute (*$name, $casesens* = `true`)

return a reference to the attribute with the name $name, if not found, return null

Parameters

string	*$name*	name of the attribute enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

Attribute found attribute

Here is the caller graph for this function:

6.12.3.21 getClock (*$name, $casesens* = `true`)

return a reference to the clock with the name $name, if not found, return null

Parameters

string	*$name*	name of the clock to search
bool	*$casesens*	take care or not of the case of the name

Returns

Clock found clock

Here is the caller graph for this function:

6.12.3.22 getComponent (*$name, $casesens* = `true`)

return a reference to the component with the name $name, if not found, return null

Parameters

string	*$name*	name of the component to search
bool	*$casesens*	take care or not of the case of the name

Returns

Component found component

6.12.3.23 getExtPort (*$name, $casesens* = `true`)

return a reference to the external port with the name $name, if not found, return null

Parameters

string	*$name*	name of the external port to search
bool	*$casesens*	take care or not of the case of the name

Returns

Port found external port

Here is the caller graph for this function:

6.12.3.24 getFile (*$name*)

return a reference to the file with the name $name, if not found, return null

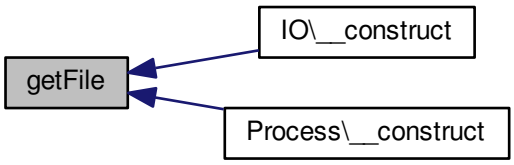
Parameters

string	*$name*	name of the file to search

Returns

File found file

Here is the caller graph for this function:



6.12.3.25 getFileByPath (*$path*)

return a reference to the file with the path $path, if not found, return null

Parameters

string	*$path*	path of the file to search

Returns

File found file

Here is the caller graph for this function:

6.12.3.26 getFlow (*$name*, *$casesens* = `true`)

return a reference to the flow with the name $name, if not found, return null

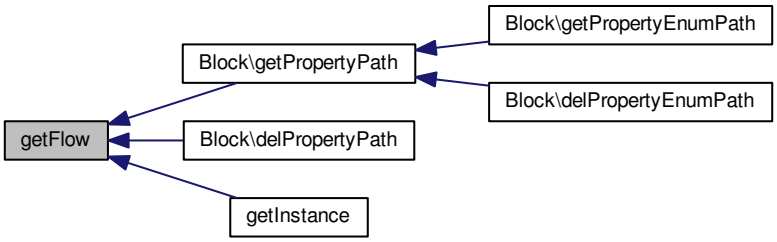
Parameters

string	*$name*	name of the flow to search
bool	*$casesens*	take care or not of the case of the name

Returns

Flow found flow

Here is the caller graph for this function:



6.12.3.27 getInfo (*$name, $casesens* = `true`)

return a reference to the component with the name $name, if not found, return null

Parameters

string	*$name*	name of the info to search
bool	*$casesens*	take care or not of the case of the name

Returns

Info found component

6.12.3.28 getInstance (*$name, $casesens* = `true`)

return a reference to the instance with the name $name, if not found, return null

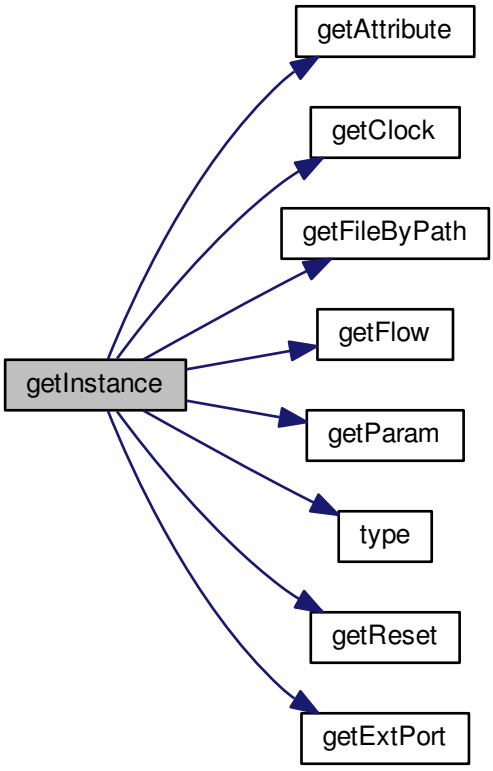
Parameters

string	*$name*	name of the instance to search
bool	*$casesens*	take care or not of the case of the name

Returns

mixed found instance

Here is the call graph for this function:



6.12.3.29 getInterface (*$name, $casesens* = `true`)

return a reference to the interface with the name $name, if not found, return null

Parameters

string	*$name*	name of the interface to search
bool	*$casesens*	take care or not of the case of the name

Returns

InterfaceBus found interface

6.12.3.30 getParam (*$name, $casesens* = `true`)

return a reference to the parameter with the name $name, if not found, return null

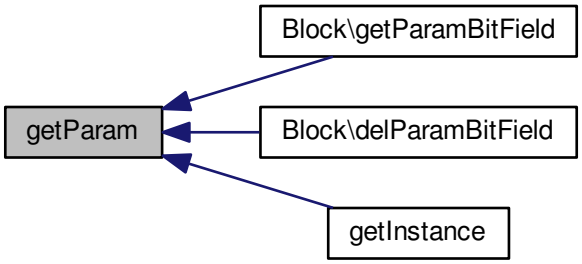
Parameters

string	*$name*	name of the parameter to search
bool	*$casesens*	take care or not of the case of the name

Returns

Param found parameter

Here is the caller graph for this function:



6.12.3.31 getPart (*$name, $casesens* = `true`)

return a reference to the part with the name $name, if not found, return null

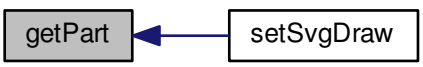
Parameters

string	*$name*	name of the component to search
bool	*$casesens*	take care or not of the case of the name

Returns

ComponentPart found component

Here is the caller graph for this function:



6.12.3.32 getReset (*$name, $casesens* = `true`)

return a reference to the reset with the name $name, if not found, return null

Parameters

string	*$name*	name of the reset to search
bool	*$casesens*	take care or not of the case of the name

Returns

Reset found reset

Here is the caller graph for this function:

6.12.3.33 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

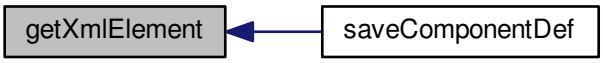
Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

Here is the caller graph for this function:



6.12.3.34 parse_xml (*$dummy* = `NULL`, *$dummy2* = `NULL`) `[protected]`

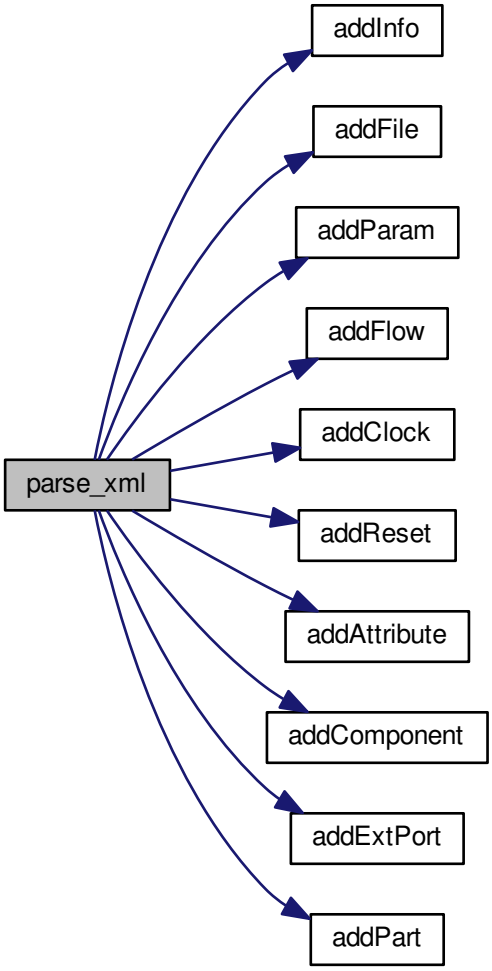
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

NULL	*$dummy*	dummy parameter to ensuze PHP7 object parent compatibility
NULL	*$dummy2*	dummy parameter to ensuze PHP7 object parent compatibility

Here is the call graph for this function:



Here is the caller graph for this function:

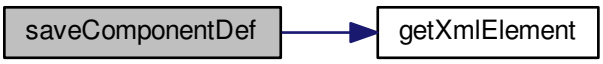
6.12.3.35 saveComponentDef (*$file*)

Helper function that save an '.comp' file.

Parameters

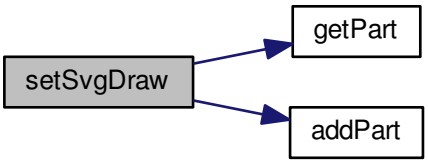
string	*$file*	file name to save

Here is the call graph for this function:



6.12.3.36 setSvgDraw (*$svgXml*, *$partName* = "main")

Here is the call graph for this function:



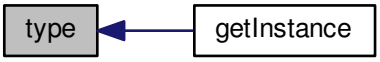
6.12.3.37 type ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

Here is the caller graph for this function:



6.12.4 Member Data Documentation

6.12.4.1 array Attribute $attributes

Array of attributes.

6.12.4.2 string $categ

Specify the categorie of the component eg : communication, imagesensor, descriptor...

6.12.4.3 array Clock $clocks

Array of clocks to drive the component.

6.12.4.4 array Component $components

Array of children components.

6.12.4.5 string $desc

Description of the flow (optional)

6.12.4.6 string $driver

Specify the name of the driver for the component.

6.12.4.7 array Port $ext_ports

Array of port abble to comunicate with the output.

6.12.4.8 array File $files

Array of files whith define the implementation of the component.

6.12.4.9 array Flow $flows

Array of flows in the component can be input flow or output.

6.12.4.10 bool $in_lib

Specify if the component is defined in the library or not.

6.12.4.11 array Info $infos

Array of information on component.

6.12.4.12 string $name

Name of the component.

6.12.4.13 array Param $params

Array of parameters class (can be Generic or dynamics parameter on BI)

6.12.4.14 Component $parentComponent

Parent components, null if the component does not have a parent.

6.12.4.15 array ComponentPart $parts

Array of graphical part.

6.12.4.16 string $path

Path where the root of files and define of the component is putted.

6.12.4.17 array Reset $resets

Array of resets, can be different type of resets.

6.12.4.18 $xml [protected]

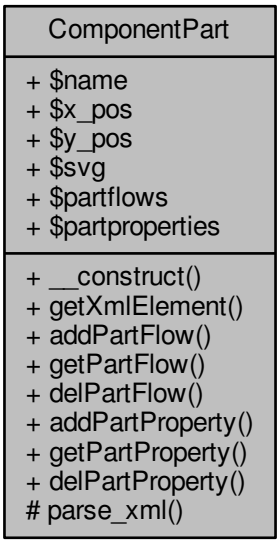
The documentation for this class was generated from the following file:

- model/component.php

6.13 ComponentPart Class Reference

ComponentPart is the the graphical definition of hardware components.

Collaboration diagram for ComponentPart:



Public Member Functions

- __construct ($xml=null)

  Constructor of the class.
- getXmlElement ($xml, $format)

  permits to output this instance
- addPartFlow ($partflow)

  Add a partflow to the param.
- getPartFlow ($name, $casesens=true)

  return a reference to the partflow with the name $name, if not found, return null
- delPartFlow ($name)

delete a partflow from his name

- addPartProperty ($partproperty)

  Add a partproperty to the param.

- getPartProperty ($name, $casesens=true)

  return a reference to the partproperty with the name $name, if not found, return null

- delPartProperty ($name)

  delete a partproperty from his name

Public Attributes

- $name

  Name of the component part.

- $x_pos

  X position on schematic (optional)

- $y_pos

  Y position on schematic (optional)

- $svg

  SVG draw for the part.

- $partflows

  Array of PartFlow.

- $partproperties

  Array.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.13.1 Detailed Description

ComponentPart is the the graphical definition of hardware components.

Multiple parts could be used to define a component with different graphical part. The default part is named `main`.

See Also

Block

6.13.2 Constructor & Destructor Documentation

6.13.2.1 __construct (*$xml* = `null`)

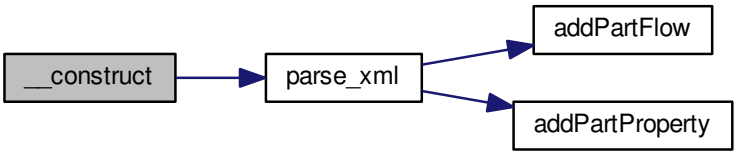
Constructor of the class.

Build an empty ComponentPart if $xml is empty, fill it with $xml if set

Parameters

SimpleXMLElement | null	*$xml*	XML element to parse if not null

Here is the call graph for this function:



6.13.3 Member Function Documentation

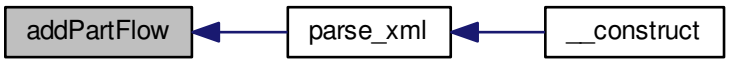
6.13.3.1 addPartFlow (*$partflow*)

Add a partflow to the param.

Parameters

PartFlow	*$partflow*	partflow to add to the param

Here is the caller graph for this function:



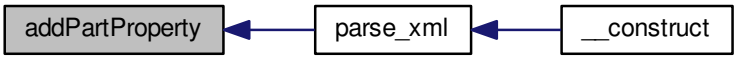
6.13.3.2 addPartProperty (*$partproperty*)

Add a partproperty to the param.

Parameters

Part-Property	*$partproperty*	partproperty to add to the param

Here is the caller graph for this function:



6.13.3.3 delPartFlow (*$name*)

delete a partflow from his name

Parameters

string	*$name*	name of the partflow to delete

6.13.3.4 delPartProperty (*$name*)

delete a partproperty from his name

Parameters

string	*$name*	name of the partproperty to delete

6.13.3.5 getPartFlow (*$name, $casesens* = `true`)

return a reference to the partflow with the name $name, if not found, return null

Parameters

string	*$name*	name of the partflow to search
bool	*$casesens*	take care or not of the case of the name

Returns

PartFlow found bitfield

6.13.3.6 getPartProperty (*$name, $casesens* = `true`)

return a reference to the partproperty with the name $name, if not found, return null

Parameters

string	*$name*	name of the partproperty to search
bool	*$casesens*	take care or not of the case of the name

Returns

PartProperty found bitfield

6.13.3.7 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.13.3.8 parse_xml (*$xml*) `[protected]`

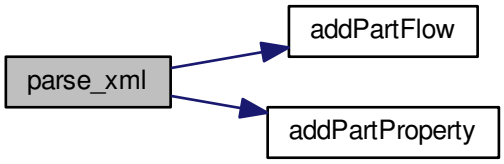
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.13.4 Member Data Documentation

6.13.4.1 string $name

Name of the component part.

6.13.4.2 array ComponentPartFlow $partflows

Array of PartFlow.

6.13.4.3 array ComponentPartProperty $partproperties

Array.

6.13.4.4 string $svg

SVG draw for the part.

6.13.4.5 int $x_pos

X position on schematic (optional)

6.13.4.6 int $y_pos

Y position on schematic (optional)

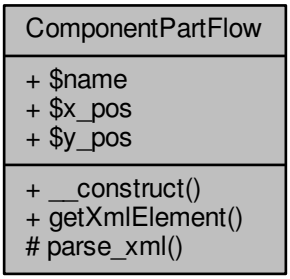
The documentation for this class was generated from the following file:

- model/componentpart.php

6.14 ComponentPartFlow Class Reference

ComponentPart is the the graphical definition of flow (position, name)

Collaboration diagram for ComponentPartFlow:



Public Member Functions

- __construct ($xml=null)

  Constructor of the class.
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the component part.
- $x_pos

  X position on schematic (optional)
- $y_pos

  Y position on schematic (optional)

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.14.1 Detailed Description

ComponentPart is the the graphical definition of flow (position, name)

Multiple part flow could be used to define a part of a component with different flow.

See Also

Block

6.14.2 Constructor & Destructor Documentation

6.14.2.1 __construct (*$xml* = `null`)

Constructor of the class.

Build an empty ComponentPart if $xml is empty, fill it with $xml if set

Parameters

SimpleXMLElement | null	*$xml*	XML element to parse if not null

Here is the call graph for this function:

6.14.3 Member Function Documentation

6.14.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.14.3.2 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.14.4 Member Data Documentation

6.14.4.1 string $name

Name of the component part.

6.14.4.2 int $x_pos

X position on schematic (optional)

6.14.4.3 int $y_pos

Y position on schematic (optional)

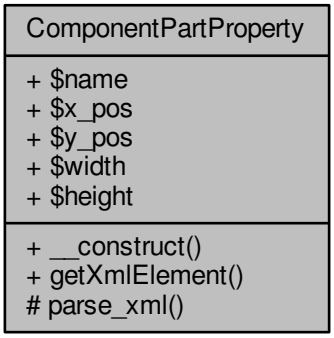
The documentation for this class was generated from the following file:

- model/componentpartflow.php

6.15 ComponentPartProperty Class Reference

ComponentPartProperty is the the graphical definition of a property (position, part)

Collaboration diagram for ComponentPartProperty:



Public Member Functions

- __construct ($xml=null)

  Constructor of the class.
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the property to draw.
- $x_pos

  X position on schematic (optional)
- $y_pos

  Y position on schematic (optional)
- $width

  width on schematic (optional)
- $height

  height on schematic (optional)

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.15.1 Detailed Description

ComponentPartProperty is the the graphical definition of a property (position, part)

Multiple part flow could be used to define a part of a component with graphical draw of property. In gpviewer, the corresponding property will be draw on the specified part with the given position and size.

See Also

ComponentPart

6.15.2 Constructor & Destructor Documentation

6.15.2.1 __construct (*$xml* = `null`)

Constructor of the class.

Build an empty ComponentPart if $xml is empty, fill it with $xml if set

Parameters

SimpleXMLElement | null	*$xml*	XML element to parse if not null

Here is the call graph for this function:

6.15.3 Member Function Documentation

6.15.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.15.3.2 parse_xml (*$xml*) `[protected]`

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.15.4 Member Data Documentation

6.15.4.1 int $height

height on schematic (optional)

6.15.4.2 string $name

Name of the property to draw.

6.15.4.3 int $width

width on schematic (optional)

6.15.4.4 int $x_pos

X position on schematic (optional)

6.15.4.5 int $y_pos

Y position on schematic (optional)

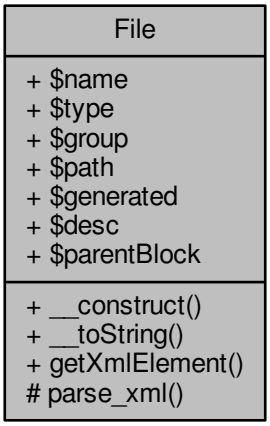
The documentation for this class was generated from the following file:

- model/componentpartproperty.php

6.16 File Class Reference

Model class to have informations about file implementation in a Component.

Collaboration diagram for File:



Public Member Functions

- __construct ($xml=null)

  constructor of File
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the file.
- $type

  File type can be : ['', 'verilog', 'vhdl', 'qip', 'sdc', 'hex'].
- $group

  Group can be : ['', 'hdl', ...].
- $path

  Sub path of the file.
- $generated

  True if this file is generated, so the path is not modified by the toolchain.
- $desc

  Description of the file (optional)
- $parentBlock

  Reference to the associated parent block.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.16.1 Detailed Description

Model class to have informations about file implementation in a Component.

They are sorted by group to give the utility of the file. File are stored in Component::$files as a list.

See Also

Component

6.16.2 Constructor & Destructor Documentation

6.16.2.1 __construct (*$xml* = `null`)

constructor of File

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.16.3 Member Function Documentation

6.16.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.16.3.2 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document

string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.16.3.3 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.16.4 Member Data Documentation

6.16.4.1 string $desc

Description of the file (optional)

6.16.4.2 bool $generated

True if this file is generated, so the path is not modified by the toolchain.

6.16.4.3 string $group

Group can be : [”, ’hdl’, ...].

6.16.4.4 string $name

Name of the file.

6.16.4.5 Block $parentBlock

Reference to the associated parent block.

6.16.4.6 string $path

Sub path of the file.

6.16.4.7 string $type

File type can be : [”, ’verilog’, ’vhdl’, ’qip’, ’sdc’, ’hex’].

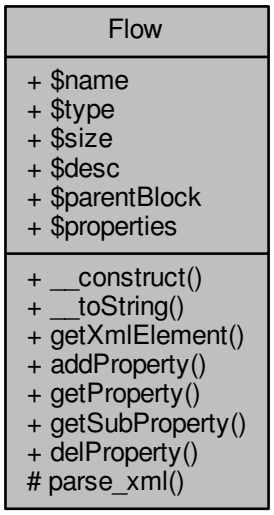
The documentation for this class was generated from the following file:

- model/file.php

6.17 Flow Class Reference

Model class to define flow hardware interface in a Component or Block.

Collaboration diagram for Flow:



Public Member Functions

- __construct ($xml=null)

  constructor of Flow
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance
- addProperty ($property)
- getProperty ($name, $casesens=true)
- getSubProperty ($name, $casesens=true)
- delProperty ($name)

Public Attributes

- $name

  Name of the flow.
- $type

  Type of the flow, can be "in" or "out".
- $size

  Size in bit of the flow.
- $desc

Description of the flow (optional)

- $parentBlock

  Reference to the associated parent block.

- $properties

  Array of property class specify the high level properties.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.17.1 Detailed Description

Model class to define flow hardware interface in a Component or Block.

Flow are stored in Component::flows as a list. A flow interface have information about the direction (in or out), the size of the data bus in bit and the name.

See Also

Component FI

6.17.2 Constructor & Destructor Documentation

6.17.2.1 __construct (*$xml* = `null`)

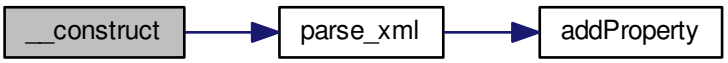
constructor of Flow

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.17.3 Member Function Documentation

6.17.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

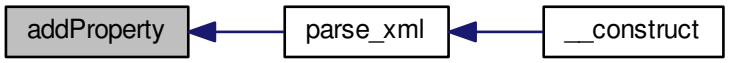
6.17.3.2 addProperty (*$property*)

Add a property to the block

Parameters

Property	*$property*	property to add to the block *

Here is the caller graph for this function:



6.17.3.3 delProperty (*$name*)

delete a property from his name

Parameters

string	*$name*	name of the property to delete *

6.17.3.4 getProperty (*$name, $casesens* = `true`)

return a reference to the property with the name $name, if not found, return null

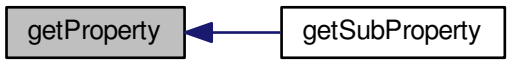
Parameters

string	*$name*	name of the property to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property *

Here is the caller graph for this function:



6.17.3.5 getSubProperty (*$name, $casesens* = `true`)

alias to getProperty($name, $casesens)

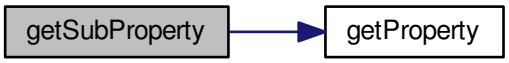
Parameters

string	*$name*	name of the property enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property ∗

Here is the call graph for this function:



6.17.3.6 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.17.3.7 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:

Here is the caller graph for this function:

6.17.4 Member Data Documentation

6.17.4.1 string $desc

Description of the flow (optional)

6.17.4.2 string $name

Name of the flow.

6.17.4.3 Block $parentBlock

Reference to the associated parent block.

6.17.4.4 array Property $properties

Array of property class specify the high level properties.

6.17.4.5 int $size

Size in bit of the flow.

6.17.4.6 string $type

Type of the flow, can be "in" or "out".

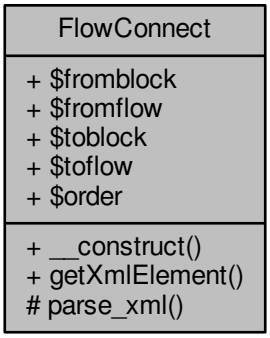
The documentation for this class was generated from the following file:

- model/flow.php

6.18 FlowConnect Class Reference

Define flow connection between two flow interface of blocks, an output flow to an input flow interface.

Collaboration diagram for FlowConnect:



Public Member Functions

- __construct ($fromBlock=NULL, $fromFlow=NULL, $toBlock=NULL, $toFlow=NULL, $order= 'msb')

  constructor of FlowConnect
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $fromblock

  Name of the block source of the flow.
- $fromflow

  Name of the flow on the block source of the flow.
- $toblock

  Name of the block sink of the flow.
- $toflow

  Name of the flow on the block sink of the flow.
- $order

  Byte ordering can be "msb" or "lsb", default value is "msb".

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.18.1 Detailed Description

Define flow connection between two flow interface of blocks, an output flow to an input flow interface.

FlowConnect are used in FI::flow_connects as a list. The set of flowconnect is used to define the flow interconnect architechture.

See Also

Block FI Flow

6.18.2 Constructor & Destructor Documentation

6.18.2.1 __construct (*$fromBlock* = NULL, *$fromFlow* = NULL, *$toBlock* = NULL, *$toFlow* = NULL, *$order* = 'msb')

constructor of FlowConnect

Parameters

SimpleXMLElement | string | null	*$fromBlock*	if it's different of null, call the xml parser to fill members
string | null	*$fromFlow*	default value contructor
string | null	*$toBlock*	default value contructor
string | null	*$toFlow*	default value contructor
string	*$order*	default value contructor

Here is the call graph for this function:

6.18.3 Member Function Documentation

6.18.3.1 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.18.3.2 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.18.4 Member Data Documentation

6.18.4.1 string $fromblock

Name of the block source of the flow.

6.18.4.2 string $fromflow

Name of the flow on the block source of the flow.

6.18.4.3 string $order

Byte ordering can be "msb" or "lsb", default value is "msb".

6.18.4.4 string $toblock

Name of the block sink of the flow.

6.18.4.5 string $toflow

Name of the flow on the block sink of the flow.

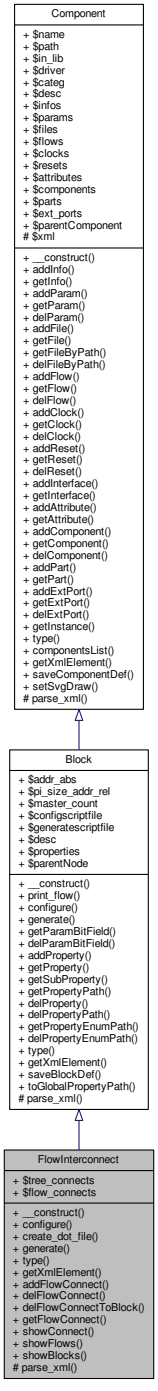
The documentation for this class was generated from the following file:

- model/flowconnect.php

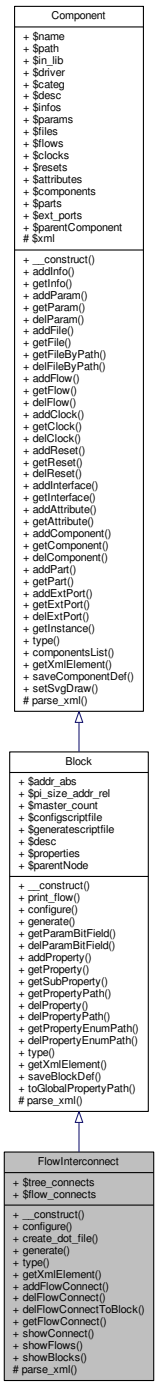
6.19 FlowInterconnect Class Reference

FlowInterconnect is the generated block to manage all the flows in the node project.

Inheritance diagram for FlowInterconnect:



Collaboration diagram for FlowInterconnect:



Public Member Functions

- __construct ()
- configure ($node, $block)
- create_dot_file ($node, $filename)
- generate ($node, $block, $path, $language)
- type ()
- getXmlElement ($xml, $format)

- addFlowConnect ($flow_connect)
- delFlowConnect ($fromBlock, $fromFlow, $toBlock, $toFlow)
- delFlowConnectToBlock ($block)
- getFlowConnect ($fromBlock, $fromFlow, $toBlock, $toFlow)

Static Public Member Functions

- static showConnect ($connect)
- static showFlows ($block, $dir)
- static showBlocks ($node)

Public Attributes

- $tree_connects
- $flow_connects

Protected Member Functions

- parse_xml ($xml=NULL, $dummy2=NULL)

Additional Inherited Members

6.19.1 Detailed Description

FlowInterconnect is the generated block to manage all the flows in the node project.

All the flow interface pass through this block. It manage all the flow connections and generate flow multiplexer in case of multiple flow out are connected to an input. The generated multiplexer, allows to dynamically change the source of the input flow. Register are created in this block to manage multiplexer.

See Also

Block Clock Parameter

6.19.2 Constructor & Destructor Documentation

6.19.2.1 __construct ( )

6.19.3 Member Function Documentation

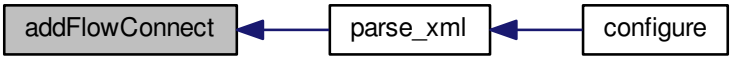
6.19.3.1 addFlowConnect (*$flow_connect*)

Add a flow connection to the block

Parameters

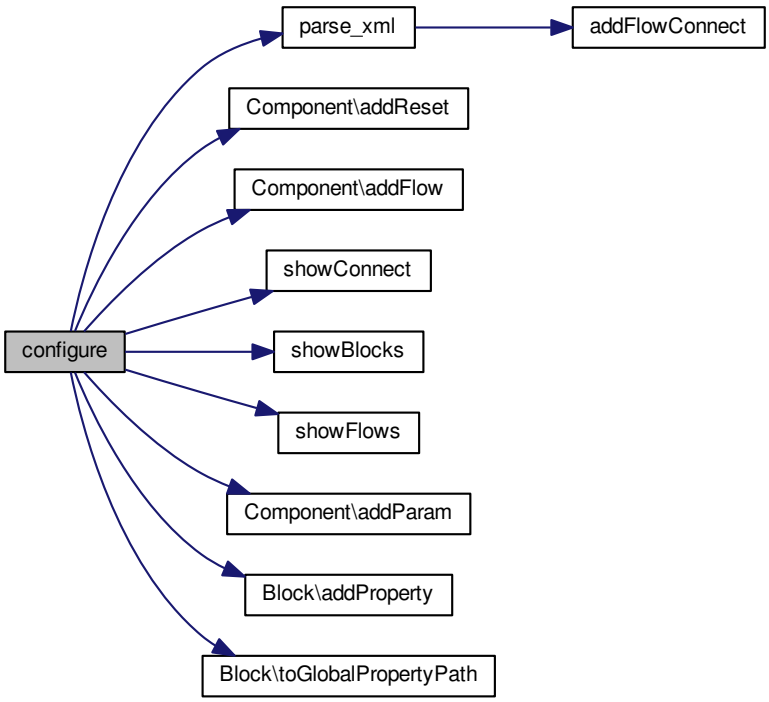
FlowConnect	*$flow_connect*	flow connection to add to the block *

Here is the caller graph for this function:



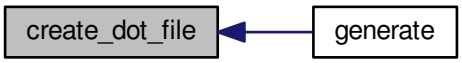
6.19.3.2 configure (*$node, $block*)

Here is the call graph for this function:



6.19.3.3 create_dot_file (*$node, $filename*)

Here is the caller graph for this function:

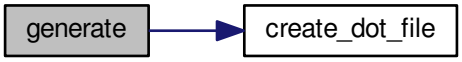


6.19.3.4 delFlowConnect (*$fromBlock, $fromFlow, $toBlock, $toFlow*)

6.19.3.5 delFlowConnectToBlock (*$block*)

6.19.3.6 generate (*$node, $block, $path, $language*)

Here is the call graph for this function:



6.19.3.7 getFlowConnect (*$fromBlock, $fromFlow, $toBlock, $toFlow*)

return a reference to the flow connection with the name $name, if not found, return false

Parameters

string	*$fromBlock*	name of the output flow block.
string	*$fromFlow*	name of the output flow in $fromBlock.
string	*$toBlock*	name of the input flow block.
string	*$toFlow*	name of the input flow block in $toBlock.

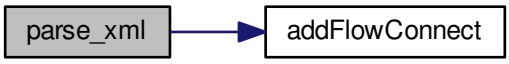
Returns

FlowConnect found flow connection *

6.19.3.8 getXmlElement (*$xml*, *$format*)

6.19.3.9 parse_xml (*$xml* = NULL, *$dummy2* = NULL) [protected]

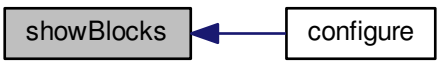
Here is the call graph for this function:



Here is the caller graph for this function:

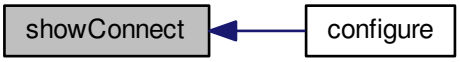
6.19.3.10 static showBlocks (*$node*) [static]

Here is the caller graph for this function:



6.19.3.11 static showConnect (*$connect*) [static]

Here is the caller graph for this function:



6.19.3.12 static showFlows (*$block, $dir*) [static]

Here is the caller graph for this function:

6.19.3.13 type ( )

6.19.4 Member Data Documentation

6.19.4.1 array FlowConnect $flow_connects

Flow connections between blocks

6.19.4.2 array TreeConnect $tree_connects

Map of computed connections, indexed by a key (blockName.flowName) with a TreeConnect object intance as value which represent the tree of flow connection

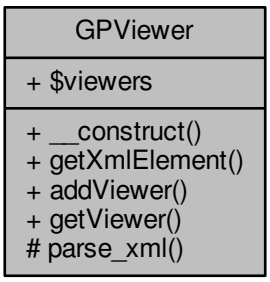
The documentation for this class was generated from the following file:

- system_interconnect/fi.php

6.20 GPViewer Class Reference

List of all the configurated viewer for gpviewer.

Collaboration diagram for GPViewer:



Public Member Functions

- __construct ($xml=NULL)

  internal function to fill this instance from input xml structure
- getXmlElement ($xml)

  permits to output this instance
- addViewer ($viewer)

  Add a viewer to the list of viewers.
- getViewer ($name)

  return a reference to the viewer with the name $name, if not found, return null

Public Attributes

- $viewers

  Array of ComConnect to give the list of all the configurated viewer for gpviewer.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.20.1 Detailed Description

List of all the configurated viewer for gpviewer.

Structure that store all gpviewer dedicated parameters. At the momement, this structure only contain the viewer list.

See Also

Viewer Node

6.20.2 Constructor & Destructor Documentation

6.20.2.1 __construct (*$xml* = NULL)

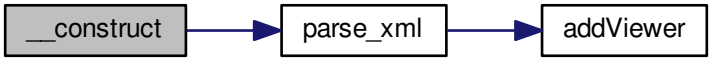
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the call graph for this function:



6.20.3 Member Function Documentation

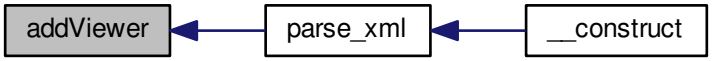
6.20.3.1 addViewer (*$viewer*)

Add a viewer to the list of viewers.

Parameters

Viewer	*$viewer*	viewer to add to the list of viewers

Here is the caller graph for this function:



6.20.3.2 getViewer (*$name*)

return a reference to the viewer with the name $name, if not found, return null

Parameters

string	*$name*	name of the viewer to search

Returns

Viewer found viewer

6.20.3.3 getXmlElement (*$xml*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document

Returns

DOMElement xml element corresponding to this current instance

6.20.3.4 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:

Here is the caller graph for this function:

6.20.4 Member Data Documentation

6.20.4.1 array Viewer $viewers

Array of ComConnect to give the list of all the configurated viewer for gpviewer.

See Also

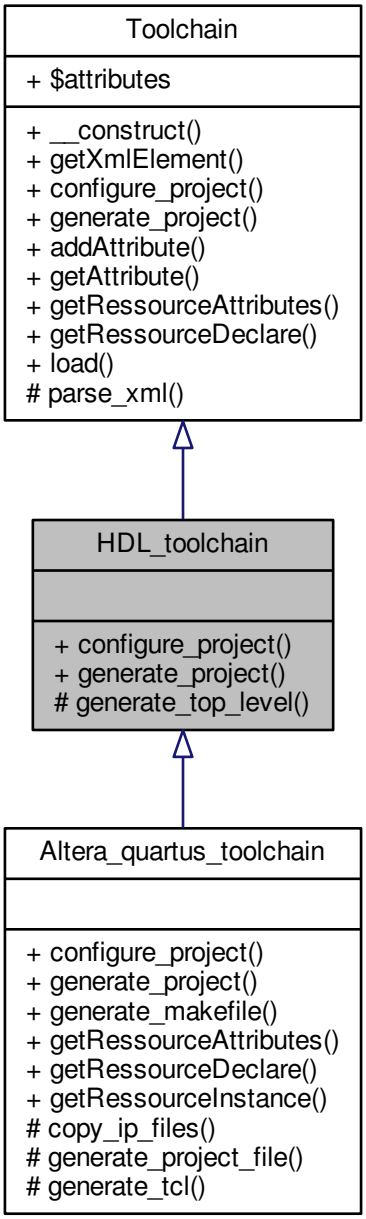
Viewer

The documentation for this class was generated from the following file:

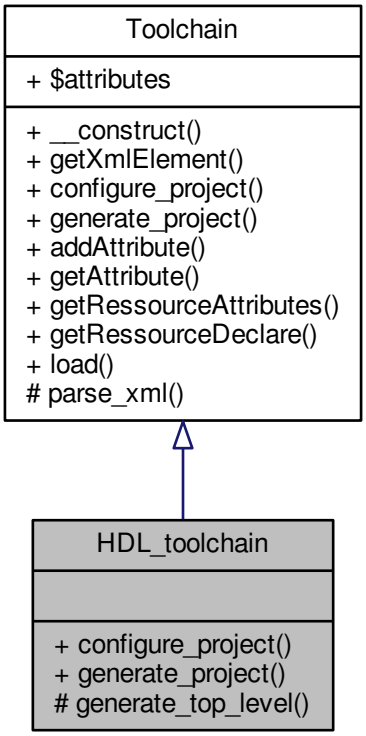
- model/gpviewer.php

6.21 HDL_toolchain Class Reference

Inheritance diagram for HDL_toolchain:



Collaboration diagram for HDL_toolchain:



Public Member Functions

- configure_project ($node)
- generate_project ($node, $path)

Protected Member Functions

- generate_top_level ($node, $path)

Additional Inherited Members

6.21.1 Member Function Documentation

6.21.1.1 configure_project (*$node*)

6.21.1.2 generate_project (*$node, $path*)

Here is the call graph for this function:

6.21.1.3 generate_top_level (*$node, $path*) [protected]

Here is the caller graph for this function:

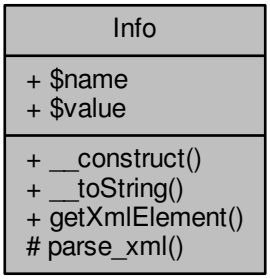
The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/hdl.php

6.22 Info Class Reference

The Info class is used to indicate information for IPs.

Collaboration diagram for Info:



Public Member Functions

- __construct ($xml=null)

  Constructor of the class.
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the info.
- $value

  Value of the info.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.22.1 Detailed Description

The Info class is used to indicate information for IPs.

Info is contained into Component class to give information for IPs like author, email, version, company, licence...

See Also

Component

6.22.2 Constructor & Destructor Documentation

6.22.2.1 __construct (*$xml* = `null`)

Constructor of the class.

Build an empty Info if $xml is empty, fill it with $xml if set

Parameters

SimpleXMLElement | null	*$xml*	XML element to parse if not null

Here is the call graph for this function:

6.22.3 Member Function Documentation

6.22.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.22.3.2 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.22.3.3 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.22.4 Member Data Documentation

6.22.4.1 string $name

Name of the info.

6.22.4.2 string $value

Value of the info.

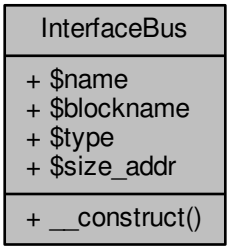
The documentation for this class was generated from the following file:

- model/info.php

6.23 InterfaceBus Class Reference

Define a bus interface for PI.

Collaboration diagram for InterfaceBus:



Public Member Functions

- __construct ($name, $blockname, $type, $size_addr)

Public Attributes

- $name

  Name of the interface.
- $blockname

  Name of the block.
- $type

  Type of connection.
- $size_addr

  Sife of the adress bus interface.

6.23.1 Detailed Description

Define a bus interface for PI.

InterfaceBus are used in Block::interfaces as a list. An interface as slave, (pi_slave) or master (pi_master), slave connection (pi_slave_conn) or master connection (pi_master_conn)

See Also

Block PI

6.23.2 Constructor & Destructor Documentation

6.23.2.1 __construct (*$name, $blockname, $type, $size_addr*)

Parameters

string	*$name*	default value contructor
string	*$blockname*	default value contructor
string	*$type*	default value contructor
int	*$size_addr*	default value contructor

6.23.3 Member Data Documentation

6.23.3.1 string $blockname

Name of the block.

6.23.3.2 string $name

Name of the interface.

6.23.3.3 int $size_addr

Sife of the adress bus interface.

6.23.3.4 string $type

Type of connection.

bi_master bi_slave bi_master_conn bi_slave_conn

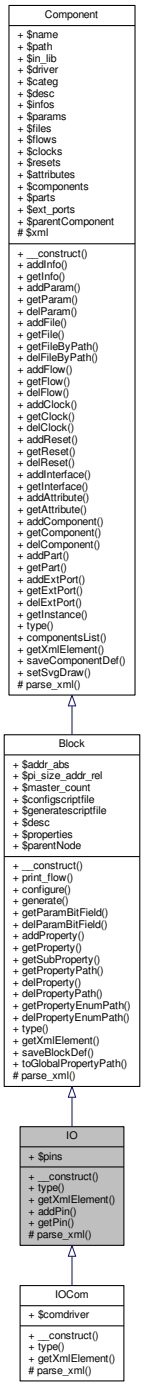
The documentation for this class was generated from the following file:

- model/interfacebus.php

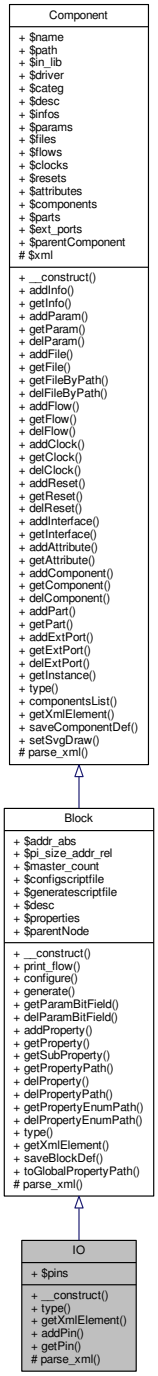
6.24 IO Class Reference

IO is the specialised implementation of Block for sensors and communication blocks.

Inheritance diagram for IO:



Collaboration diagram for IO:



Public Member Functions

- __construct ($io_device_element=null, $io_node_element=null)

  constructor of IO
- type ()

  Returns the type of the block as string, redefined by children.
- getXmlElement ($xml, $format)

permits to output this instance

- addPin ($pin)

  Add a pin to the block.

- getPin ($name, $casesens=true)

  return a reference to the pin with the name $name, if not found, return null

Public Attributes

- $pins

  External pins mapping for blocks abble to comunicate with the output.

Protected Member Functions

- parse_xml ($io_device_element=NULL, $io_node_element=NULL)

  internal function to fill this instance from input xml structure

Additional Inherited Members

6.24.1 Detailed Description

IO is the specialised implementation of Block for sensors and communication blocks.

In addition to simple Block, it allows the use of pins and external port to declare physical interface.

It only contains the list of :

- pins and ports as external io IO::$pins, IO::$ext_ports

See Also

Block

6.24.2 Constructor & Destructor Documentation

6.24.2.1 __construct (*$io_device_element* = `null`, *$io_node_element* = `null`)

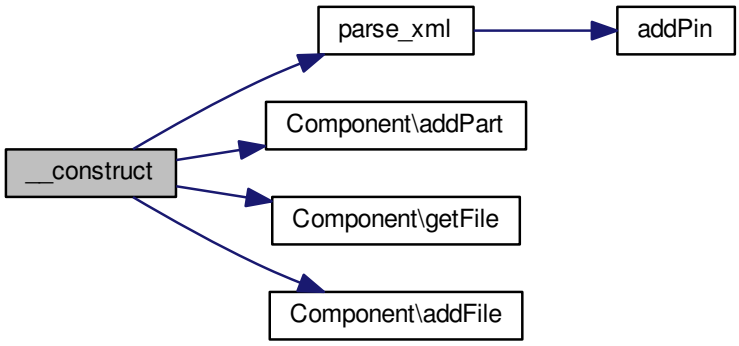
constructor of IO

Initialise all the internal members and call parse_xml if $io_device_element is an SimpleXMLElement object. Else, it open the file with the path $io_device_element as string or the io with the name $io_device_element in library

Parameters

SimpleXMLElement | string | null	*$io_device_element*	if it's different of null, call the xml parser to fill members. In case of string type, loads the IO in lib from the name or from the path in project.
SimpleXMLElement | null	*$io_node_element*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.24.3 Member Function Documentation

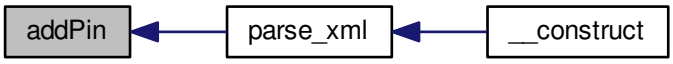
6.24.3.1 addPin (*$pin*)

Add a pin to the block.

Parameters

Pin	*$pin*	pin to add to the block

Here is the caller graph for this function:



6.24.3.2 getPin (*$name, $casesens* = `true`)

return a reference to the pin with the name $name, if not found, return null

Parameters

string	*$name*	name of the pin to search
bool	*$casesens*	take care or not of the case of the name

Returns

Pin found pin

6.24.3.3 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.24.3.4 parse_xml (*$io_device_element* = NULL, *$io_node_element* = NULL) [protected]

internal function to fill this instance from input xml structure

Parameters

SimpleXM-LElement	*$io_device_-element*	element from io in lib
SimpleXM-LElement	*$io_node_-element*	element from the node

Here is the call graph for this function:

Here is the caller graph for this function:

6.24.3.5 type ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

6.24.4 Member Data Documentation

6.24.4.1 array Pin $pins

External pins mapping for blocks abble to comunicate with the output.

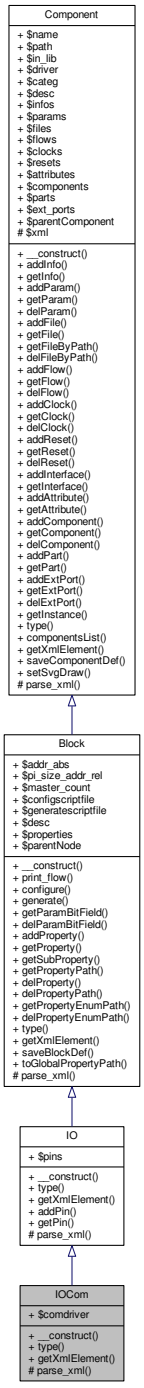
The documentation for this class was generated from the following file:

- model/io.php

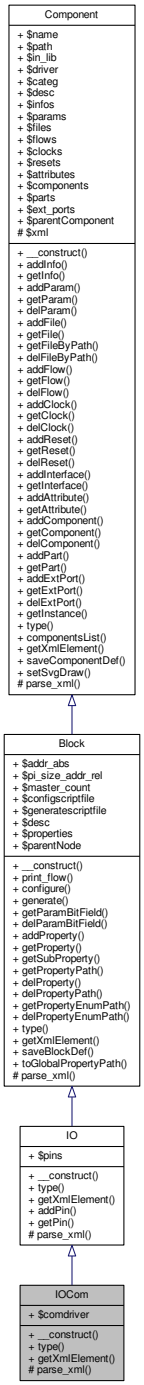
6.25 IOCom Class Reference

IOCom is the specialised implementation of IO.

Inheritance diagram for IOCom:



Collaboration diagram for IOCom:



Public Member Functions

- __construct ($io_device_element, $io_node_element=null)

  constructor of IO
- type ()

  Returns the type of the block as string, redefined by children.
- getXmlElement ($xml, $format)

permits to output this instance

Public Attributes

- $comdriver

Protected Member Functions

- parse_xml ($io_device_element=NULL, $io_node_element=NULL)
  internal function to fill this instance from input xml structure

Additional Inherited Members

6.25.1 Detailed Description

IOCom is the specialised implementation of IO.

It allows the use of communication links and protocol declaration.

See Also

IO ComConnect

6.25.2 Constructor & Destructor Documentation

6.25.2.1 __construct (*$io_device_element, $io_node_element* = `null`)

constructor of IO

Initialise all the internal members and call parse_xml if $io_device_element is an SimpleXMLElement object. Else, it open the file with the path $io_device_element as string or the io with the name $io_device_element in library

Parameters

SimpleXMLElement | string | null	*$io_device_element*	if it's different of null, call the xml parser to fill members
SimpleXMLElement | null	*$io_node_element*	if it's different of null, call the xml parser to fill members

6.25.3 Member Function Documentation

6.25.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document

string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.25.3.2 parse_xml (*$io_device_element* = NULL, *$io_node_element* = NULL) [protected]

internal function to fill this instance from input xml structure

Parameters

SimpleXMLElement	*$io_device_element*	element from io in lib
SimpleXMLElement	*$io_node_element*	element from the node

6.25.3.3 type ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

6.25.4 Member Data Documentation

6.25.4.1 $comdriver

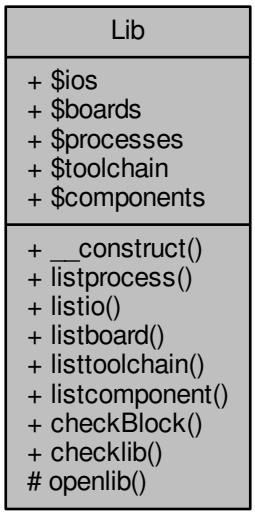
The documentation for this class was generated from the following file:

- model/iocom.php

6.26 Lib Class Reference

Lib is a container that store all the IPs available in library path.

Collaboration diagram for Lib:



Public Member Functions

- __construct ($libpaths)

  Constructor of Lib from an array of library path.
- listprocess ()

  Prints the list of process (only the name)
- listio ()

  Prints the list of devices (only the name)
- listboard ()

  Prints the list of board (only the name)
- listtoolchain ()

  Prints the list of toolchain (only the name)
- listcomponent ()

  Prints the list of component (only the name)
- checkBlock ($block)

  Try to open all the process.
- checklib ()

  Try to open all the process.

Public Attributes

- $ios

  Array of IOs name available in library.
- $boards

  Array of boards name available in library.
- $processes
- $toolchain

Array of toolchains name available in library.

- $components

  Array of components name available in library.

Protected Member Functions

- openlib ($libpath)

  Opens the path and search all the available IP in all of them.

6.26.1 Detailed Description

Lib is a container that store all the IPs available in library path.

See Also

Board IO Process Toolchain

6.26.2 Constructor & Destructor Documentation

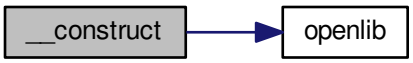
6.26.2.1 __construct (*$libpaths*)

Constructor of Lib from an array of library path.

Parameters

array | string	*$libpaths*	

Here is the call graph for this function:

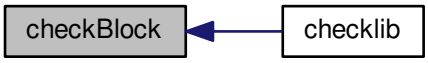


6.26.3 Member Function Documentation

6.26.3.1 checkBlock (*$block*)

Try to open all the process.

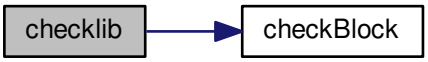
Here is the caller graph for this function:



6.26.3.2 checklib ( )

Try to open all the process.

Here is the call graph for this function:



6.26.3.3 listboard ( )

Prints the list of board (only the name)

6.26.3.4 listcomponent ( )

Prints the list of component (only the name)

6.26.3.5 listio ( )

Prints the list of devices (only the name)

6.26.3.6 listprocess ( )

Prints the list of process (only the name)

6.26.3.7 listtoolchain ( )

Prints the list of toolchain (only the name)

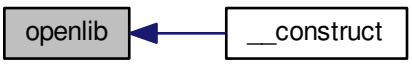
6.26.3.8 openlib (*$libpath*) [protected]

Opens the path and search all the available IP in all of them.

Parameters

array | string	*$libpath*	Array of library path

Here is the caller graph for this function:



6.26.4 Member Data Documentation

6.26.4.1 array LibItem $boards

Array of boards name available in library.

6.26.4.2 array LibItem $components

Array of components name available in library.

6.26.4.3 array LibItem $ios

Array of IOs name available in library.

6.26.4.4 $processes

6.26.4.5 array LibItem $toolchain

Array of toolchains name available in library.

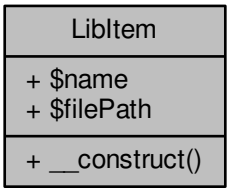
The documentation for this class was generated from the following file:

- model/lib.php

6.27 LibItem Class Reference

LibItem is an item of the Lib container.

Collaboration diagram for LibItem:



Public Member Functions

- __construct ($name, $filePath)
  Constructor of a LibItem.

Public Attributes

- $name
  name of LibItem
- $filePath
  file path of the LibItem containing file, path and extension.

6.27.1 Detailed Description

LibItem is an item of the Lib container.

See Also

Lib Board IO Process Toolchain

6.27.2 Constructor & Destructor Documentation

6.27.2.1 __construct (*$name, $filePath*)

Constructor of a LibItem.

Parameters

string	*$name*	name of LibItem
string	*$filePath*	file path of the LibItem containing file, path and extension.

6.27.3 Member Data Documentation

6.27.3.1 string $filePath

file path of the LibItem containing file, path and extension.

6.27.3.2 string $name

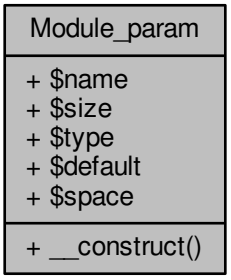
name of LibItem

The documentation for this class was generated from the following file:

- model/libitem.php

6.28 Module_param Class Reference

Collaboration diagram for Module_param:



Public Member Functions

- __construct ($name= ”, $size=1, $type= ”, $default= ”, $space=false)

Public Attributes

- $name

- $size
- $type
- $default
- $space

6.28.1 Constructor & Destructor Documentation

6.28.1.1 __construct (*$name* = '', *$size* = 1, *$type* = '', *$default* = '', *$space* = false)

6.28.2 Member Data Documentation

6.28.2.1 $default

6.28.2.2 $name

6.28.2.3 $size

6.28.2.4 $space

6.28.2.5 $type

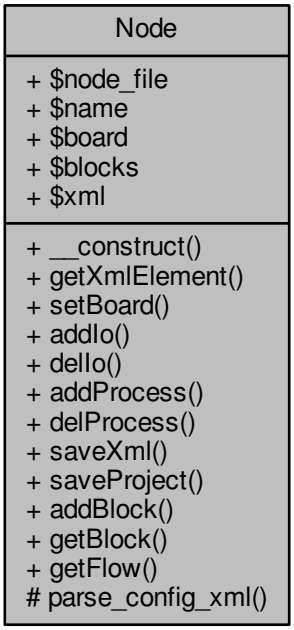
The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/vhdl_generator.php

6.29 Node Class Reference

Node is the base class container that store all the configuration of a node.

Collaboration diagram for Node:



Public Member Functions

- __construct ($node_file=null)
  constructor of Node
- getXmlElement ($xml, $format)
  permits to output this instance
- setBoard ($boardName)
  Sets the board from his name in library.
- addIo ($ioName)
  Adds an IO from his name or path.
- delIo ($ioName)
  Delete an IO block from his name.
- addProcess ($processName, $processDriver)
  Adds a process from his name or path.
- delProcess ($processName)
  Delete a process block from his name.
- saveXml ($file)
  Save the whole structure with all data including all internal children data.
- saveProject ($file)
  Save the project with the minimal information needed.
- addBlock ($block)
  Add a block to the node.
- getBlock ($name)
  return a reference to the block with the name $name, if not found, return null
- getFlow ($name)
  return a reference to the flow with the name $name, if not found, return null

Public Attributes

- $node_file
  The complete path of the file of the input definition.
- $name
  Name of the node.
- $board
  Board structure of the node.
- $blocks
  Array of al the blocks (process or io) contain in the node.
- $xml

Protected Member Functions

- parse_config_xml ($node_file)
  internal function to fill this instance from input xml file

6.29.1 Detailed Description

Node is the base class container that store all the configuration of a node.

A node in GPStudio is typically a smart camera or a server. It store the list of blocks and the local configuration of the node with the definition of the board.

See Also

Block Board

6.29.2 Constructor & Destructor Documentation

6.29.2.1 __construct (*$node_file* = `null`)

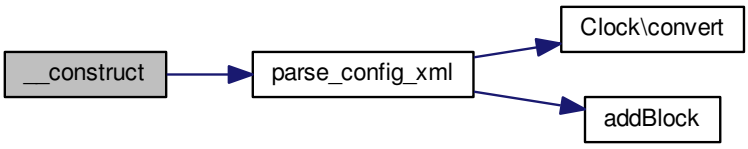
constructor of Node

Initialise all the internal members and call parse_config_xml if $node_file is set

Parameters

string	*$node_file*	Node file name

Here is the call graph for this function:



6.29.3 Member Function Documentation

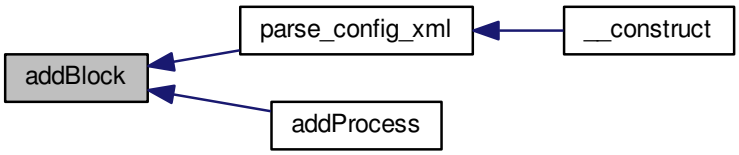
6.29.3.1 addBlock (*$block*)

Add a block to the node.

Parameters

Block	*$block*	block to add to the node

Here is the caller graph for this function:



6.29.3.2 addIo (*$ioName*)

Adds an IO from his name or path.

Gets the IO block from the library or local project and add it to the node.

Parameters

string	*$ioName*	name or path of the io to add

6.29.3.3 addProcess (*$processName, $processDriver*)

Adds a process from his name or path.

Gets the process block from the library or local project and add it to the node.

Parameters

string	*$processName*	name or path of the io to add
string	*$processDriver*	driver name or path of the process to add

Here is the call graph for this function:

6.29.3.4 delIo (*$ioName*)

Delete an IO block from his name.

Parameters

string	*$ioName*	name of the io block to delete

6.29.3.5 delProcess (*$processName*)

Delete a process block from his name.

Parameters

string	*$processName*	name of the process block to delete

6.29.3.6 getBlock (*$name*)

return a reference to the block with the name $name, if not found, return null

Parameters

string	*$name*	name of the block to search

Returns

Block found block

Here is the caller graph for this function:

6.29.3.7 getFlow (*$name*)

return a reference to the flow with the name $name, if not found, return null

Parameters

string	*$name*	name of the flow to search composed by the name of the block followed by the name of the flow. Ex : 'block1.flowin0'

Returns

Flow found flow

Here is the call graph for this function:

6.29.3.8 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

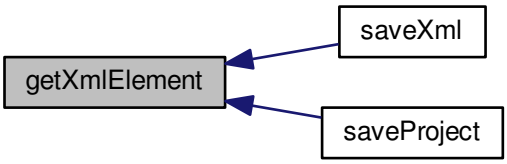
Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

Here is the caller graph for this function:



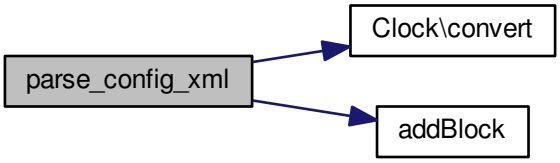
6.29.3.9 parse_config_xml (*$node_file*) [protected]

internal function to fill this instance from input xml file

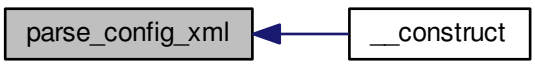
Parameters

string	*$node_file*	Node file name

Here is the call graph for this function:



Here is the caller graph for this function:



6.29.3.10 saveProject (*$file*)

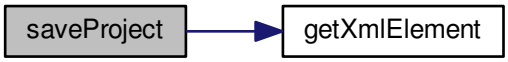
Save the project with the minimal information needed.

Generally used to save the '.node' project file.

Parameters

string	*$file*	file name to save

Here is the call graph for this function:



6.29.3.11 saveXml (*$file*)

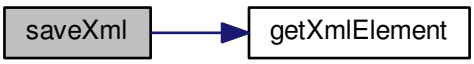
Save the whole structure with all data including all internal children data.

Generally used to save the 'node_generated.xml' file in build dir.

Parameters

string	*$file*	file name to save

Here is the call graph for this function:



6.29.3.12 setBoard (*$boardName*)

Sets the board from his name in library.

Gets the Board block from the library or local project and add it to the node.

Parameters

string	*$boardName*	name of the board.

See Also

Board

6.29.4 Member Data Documentation

6.29.4.1 array Block $blocks

Array of al the blocks (process or io) contain in the node.

6.29.4.2 Board $board

Board structure of the node.

6.29.4.3 string $name

Name of the node.

6.29.4.4 string $node_file

The complete path of the file of the input definition.

6.29.4.5 $xml

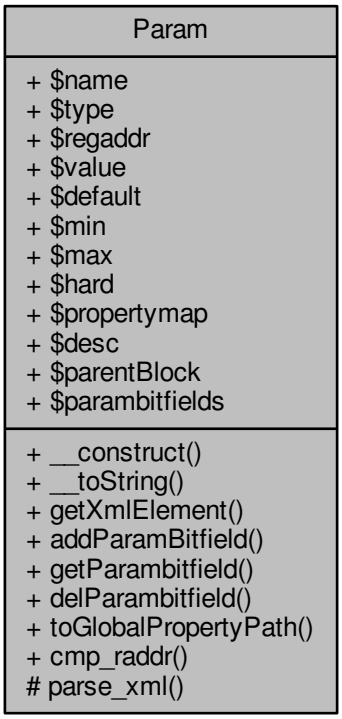
The documentation for this class was generated from the following file:

- model/node.php

6.30 Param Class Reference

Param handle a constant parameter (generic for VHDL, param for verilog constant for C/C++) or register for hardware implementation.

Collaboration diagram for Param:



Public Member Functions

- __construct ($xml=null)

- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance
- addParamBitfield ($parambitfield)

  Add a parambitfield to the param.
- getParambitfield ($name, $casesens=true)

  return a reference to the bitfield with the name $name, if not found, return null
- delParambitfield ($name)

  delete a bitfield from his name
- toGlobalPropertyPath ($blockName)

  Redefines all the contained properties to the global context by adding the name of the block at the begining of all of properties map.

Static Public Member Functions

- static cmp_raddr ($a, $b)

  Compare two param object address to sort it by address.

Public Attributes

- $name

  Name of the param.
- $type
- $regaddr

  Relative adress of the param.
- $value

  Current value of the param.
- $default

  Default value of the param.
- $min

  Minimum value of the param.
- $max

  Maximum value of the param.
- $hard

  Generic parameter (hard = true) or dynamic one (hard = false)
- $propertymap

  Mapping to properties (optional)
- $desc

  Description of the param (optional)
- $parentBlock

  Reference to the associated parent block.
- $parambitfields

  Array of bitfields if param contain different bitfield (optional)

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.30.1 Detailed Description

Param handle a constant parameter (generic for VHDL, param for verilog constant for C/C++) or register for hardware implementation.

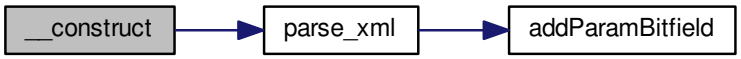
See Also

Block

6.30.2 Constructor & Destructor Documentation

6.30.2.1 __construct (*$xml* = `null`)

Here is the call graph for this function:



6.30.3 Member Function Documentation

6.30.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

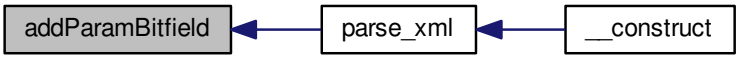
6.30.3.2 addParamBitfield (*$parambitfield*)

Add a parambitfield to the param.

Parameters

Param-Bitfield	*$parambitfield*	parambitfield to add to the param

Here is the caller graph for this function:



6.30.3.3 static cmp_raddr (*$a*, *$b*) `[static]`

Compare two param object address to sort it by address.

Parameters

Param	*$a*	first Param object to compare
Param	*$b*	second Param object to compare

Returns

int -1, 1 or 0 to say if it is smaller, bigger or equal

6.30.3.4 delParambitfield (*$name*)

delete a bitfield from his name

Parameters

string	*$name*	name of the bitfield to delete

6.30.3.5 getParambitfield (*$name, $casesens* = `true`)

return a reference to the bitfield with the name $name, if not found, return null

Parameters

string	*$name*	name of the parambitfield to search
bool	*$casesens*	take care or not of the case of the name

Returns

ParamBitfield found bitfield

6.30.3.6 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.30.3.7 parse_xml (*$xml*) `[protected]`

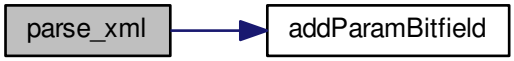
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.30.3.8 toGlobalPropertyPath (*$blockName*)

Redefines all the contained properties to the global context by adding the name of the block at the begining of all of properties map.

Parameters

string	*$blockName*	Name of of the block to append before all the propertymap expression

Here is the call graph for this function:

6.30.4 Member Data Documentation

6.30.4.1 string $default

Default value of the param.

6.30.4.2 string $desc

Description of the param (optional)

6.30.4.3 bool $hard

Generic parameter (hard = true) or dynamic one (hard = false)

6.30.4.4 string $max

Maximum value of the param.

6.30.4.5 string $min

Minimum value of the param.

6.30.4.6 string $name

Name of the param.

6.30.4.7 array ParamBitfield $parambitfields

Array of bitfields if param contain different bitfield (optional)

6.30.4.8 Block $parentBlock

Reference to the associated parent block.

6.30.4.9 string $propertymap

Mapping to properties (optional)

6.30.4.10 int $regaddr

Relative adress of the param.

6.30.4.11 string $type

6.30.4.12 string $value

Current value of the param.

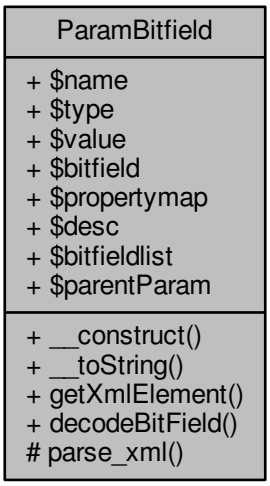
The documentation for this class was generated from the following file:

- model/param.php

6.31 ParamBitfield Class Reference

Bit field for param when param are registers.

Collaboration diagram for ParamBitfield:



Public Member Functions

- __construct ($xml=null)

  constructor of ParamBitfield
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Static Public Member Functions

- static decodeBitField ($string)

  bit field expression parser that gives the list of bits

Public Attributes

- $name

  Name of the bitfield.
- $type

  Type of the value.
- $value

  Current value of the bitfield.
- $bitfield

  Bitfield expression in text format, example: "2-0" "2,6,1" "15-4,0".
- $propertymap

  Mapping to properties (optional)
- $desc

  Description of the bitfield (optional)

- $bitfieldlist
  Array of int for concerned bit in the param.
- $parentParam
  Reference to the associated parent Param.

Protected Member Functions

- parse_xml ($xml)
  internal function to fill this instance from input xml structure

6.31.1 Detailed Description

Bit field for param when param are registers.

A param can contains multiple bitfieds and each bit field can be composed by one or more bits.

See Also

Param

6.31.2 Constructor & Destructor Documentation

6.31.2.1 __construct (*$xml* = `null`)

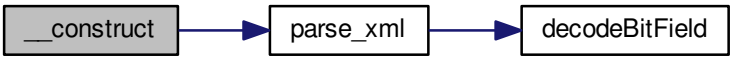
constructor of ParamBitfield

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.31.3 Member Function Documentation

6.31.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.31.3.2 static decodeBitField (*$string*) `[static]`

bit field expression parser that gives the list of bits

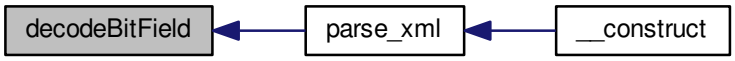
Parameters

type	*$string*	bit field expression

Returns

array list of all bits ordered

Here is the caller graph for this function:



6.31.3.3 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.31.3.4 parse_xml (*$xml*) [protected]

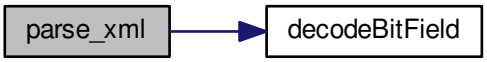
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.31.4 Member Data Documentation

6.31.4.1 string $bitfield

Bitfield expression in text format, example: "2-0" "2,6,1" "15-4,0".

6.31.4.2 array int $bitfieldlist

Array of int for concerned bit in the param.

6.31.4.3 string $desc

Description of the bitfield (optional)

6.31.4.4 string $name

Name of the bitfield.

6.31.4.5 Param $parentParam

Reference to the associated parent Param.

6.31.4.6 string $propertymap

Mapping to properties (optional)

6.31.4.7 string $type

Type of the value.

6.31.4.8 string $value

Current value of the bitfield.

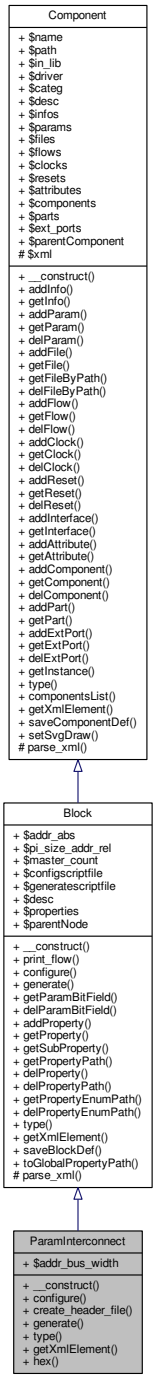
The documentation for this class was generated from the following file:

- model/parambitfield.php

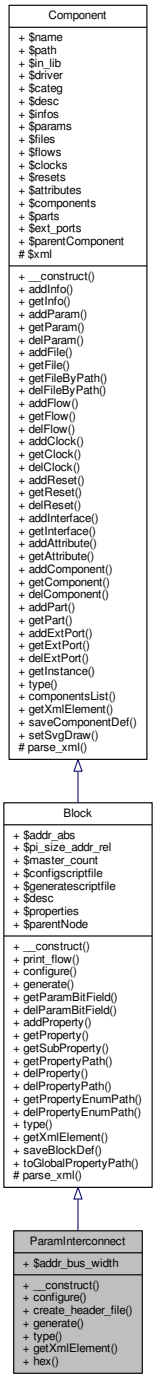
6.32 ParamInterconnect Class Reference

ParamInterconnect is the generated block to manage all the parameter interfaces.

Inheritance diagram for ParamInterconnect:



Collaboration diagram for ParamInterconnect:



Public Member Functions

- __construct ()
- configure ($node, $block)
- create_header_file ($node, $filename)
- generate ($node, $block, $path, $language)
- type ()
- getXmlElement ($xml, $format)

Static Public Member Functions

- static hex ($value, $width)

Public Attributes

- $addr_bus_width

Additional Inherited Members

6.32.1 Detailed Description

ParamInterconnect is the generated block to manage all the parameter interfaces.

It internally contain bus interconnect to all block that have a PI interface. The generation of the block choose adress for each block.

See Also

Block Parameter

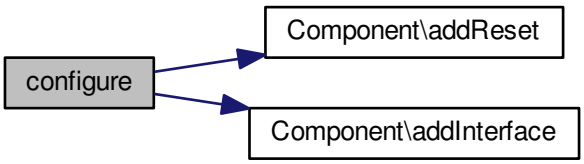
6.32.2 Constructor & Destructor Documentation

6.32.2.1 __construct ( )

6.32.3 Member Function Documentation

6.32.3.1 configure (*$node, $block*)

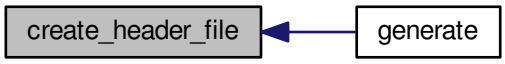
Here is the call graph for this function:



6.32.3.2 create_header_file (*$node, $filename*)

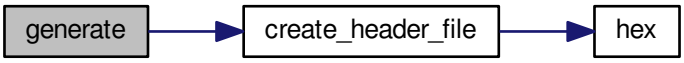
Here is the call graph for this function:

Here is the caller graph for this function:



6.32.3.3 generate (*$node, $block, $path, $language*)

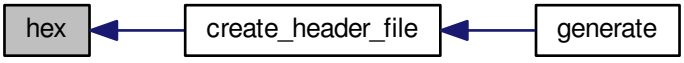
Here is the call graph for this function:



6.32.3.4 getXmlElement (*$xml, $format*)

6.32.3.5 static hex (*$value, $width*) [static]

Here is the caller graph for this function:



6.32.3.6 type ( )

6.32.4 Member Data Documentation

6.32.4.1 $addr_bus_width

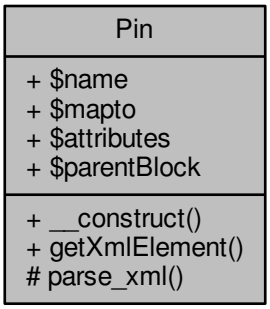
The documentation for this class was generated from the following file:

- system_interconnect/pi.php

6.33 Pin Class Reference

Pin is the physical mapping between external port of an IO block and chip pins.

Collaboration diagram for Pin:



Public Member Functions

- __construct ($xml=null)

  constructor of Pin
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Pin name to map.
- $mapto

  Name of the external pin to map.
- $attributes

  Array of attributes of the pin (optional)
- $parentBlock

  Reference to the associated parent block.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.33.1 Detailed Description

Pin is the physical mapping between external port of an IO block and chip pins.

See Also

IO Board Port

6.33.2 Constructor & Destructor Documentation

6.33.2.1 __construct (*$xml* = `null`)

constructor of Pin

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.33.3 Member Function Documentation

6.33.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.33.3.2 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.33.4 Member Data Documentation

6.33.4.1 array Attribute $attributes

Array of attributes of the pin (optional)

6.33.4.2 string $mapto

Name of the external pin to map.

6.33.4.3 string $name

Pin name to map.

6.33.4.4 Block $parentBlock

Reference to the associated parent block.

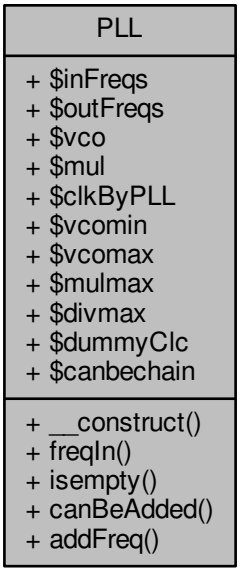
The documentation for this class was generated from the following file:

- model/pin.php

6.34 PLL Class Reference

PLL is a conveniant system to help CI PLL assignation and computation.

Collaboration diagram for PLL:



Public Member Functions

- __construct ()
- freqIn ()
- isempty ()
- canBeAdded ($clock)
- addFreq ($clock)

Public Attributes

- $inFreqs
- $outFreqs
- $vco
- $mul
- $clkByPLL
- $vcomin
- $vcomax
- $mulmax
- $divmax
- $dummyClc
- $canbechain

6.34.1 Detailed Description

PLL is a conveniant system to help CI PLL assignation and computation.

See Also

IO Board Port

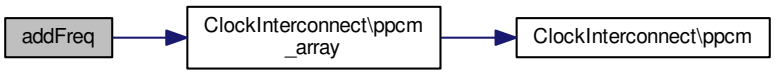
6.34.2 Constructor & Destructor Documentation

6.34.2.1 __construct ( )

6.34.3 Member Function Documentation

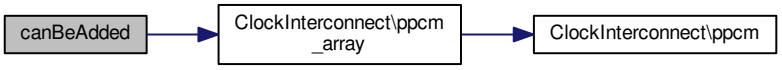
6.34.3.1 addFreq (*$clock*)

Here is the call graph for this function:



6.34.3.2 canBeAdded (*$clock*)

Here is the call graph for this function:



6.34.3.3 freqIn ( )

6.34.3.4 isempty ( )

6.34.4 Member Data Documentation

6.34.4.1 $canbechain

6.34.4.2 $clkByPLL

6.34.4.3 $divmax

6.34.4.4 $dummyClc

6.34.4.5 $inFreqs

6.34.4.6 $mul

6.34.4.7 $mulmax

6.34.4.8 $outFreqs

6.34.4.9 $vco

6.34.4.10 $vcomax

6.34.4.11 $vcomin

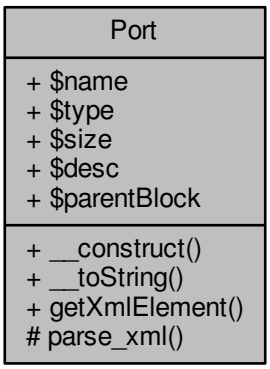
The documentation for this class was generated from the following file:

- model/pll.php

6.35 Port Class Reference

Port is external port definition for IO block.

Collaboration diagram for Port:



Public Member Functions

- __construct ($xml=null)

  constructor of Port
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the interface.
- $type

  Type of the port.
- $size

  Size of the port in bit.
- $desc

  Description of the port (optional)

- $parentBlock

  Reference to the associated parent block.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.35.1 Detailed Description

Port is external port definition for IO block.

The relation between port and physical pin mapping is ensurred by pin in board definition.

See Also

IO Board Pin

6.35.2 Constructor & Destructor Documentation

6.35.2.1 __construct (*$xml* = `null`)

constructor of Port

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.35.3 Member Function Documentation

6.35.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.35.3.2 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.35.3.3 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.35.4 Member Data Documentation

6.35.4.1 string $desc

Description of the port (optional)

6.35.4.2 string $name

Name of the interface.

6.35.4.3 Block $parentBlock

Reference to the associated parent block.

6.35.4.4 int $size

Size of the port in bit.

6.35.4.5 string $type

Type of the port.

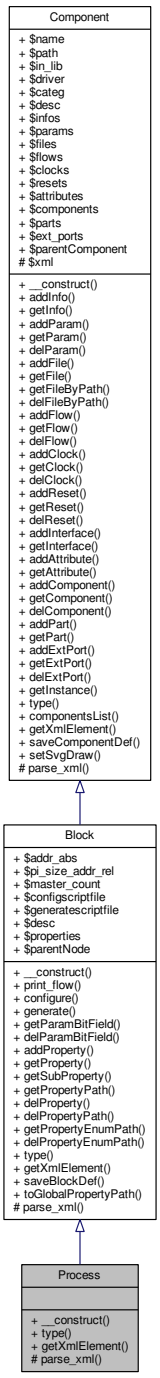
The documentation for this class was generated from the following file:

- model/port.php

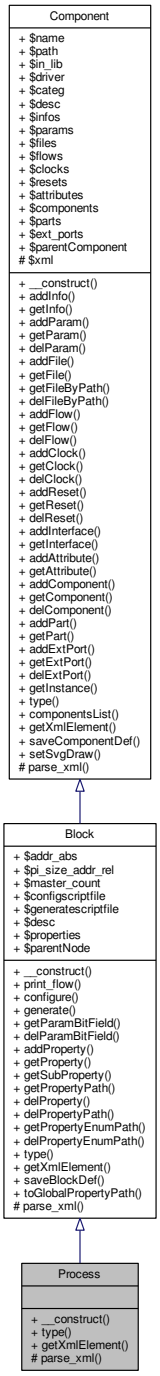
6.36 Process Class Reference

Process is the specialised implementation of Block for processes.

Inheritance diagram for Process:



Collaboration diagram for Process:



Public Member Functions

- __construct ($process_node_element=NULL)

  constructor of Process
- type ()

  Returns the type of the block as string, redefined by children.
- getXmlElement ($xml, $format)

permits to output this instance

Protected Member Functions

- parse_xml ($process_node_element=NULL, $dummy=NULL)

  internal function to fill this instance from input xml structure

Additional Inherited Members

6.36.1 Detailed Description

Process is the specialised implementation of Block for processes.

Compared to Block, this class does not contain additionnal members. It just contains library parser and helper for process. A process contain at least one input flow and one output flow.

See Also

Block

6.36.2 Constructor & Destructor Documentation

6.36.2.1 __construct (*$process_node_element* = `NULL`)

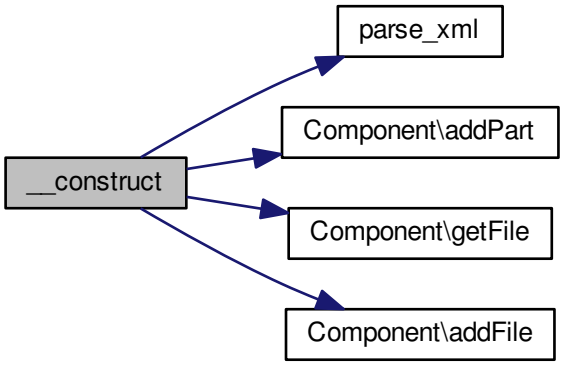
constructor of Process

Initialise all the internal members and call parse_xml if $process_node_element is an SimpleXMLElement object. Else, it open the file with the path $process_node_element as string or the process with the name $process_node_element in library

Parameters

SimpleXMLElement | string | null	*$process_node_element*	if it's different of null, call the xml parser to fill members In case of string type, loads the process in lib from the name or from the path in project.

Here is the call graph for this function:



6.36.3 Member Function Documentation

6.36.3.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.36.3.2 parse_xml (*$process_node_element* = `NULL`, *$dummy* = `NULL`) `[protected]`

internal function to fill this instance from input xml structure

Parameters

SimpleXM-LElement	*$process_node-_element*	element from io in lib
NULL	*$dummy*	dummy parameter to ensuze PHP7 object parent compatibility

Here is the caller graph for this function:

6.36.3.3 type ( )

Returns the type of the block as string, redefined by children.

Returns

string type of the block.

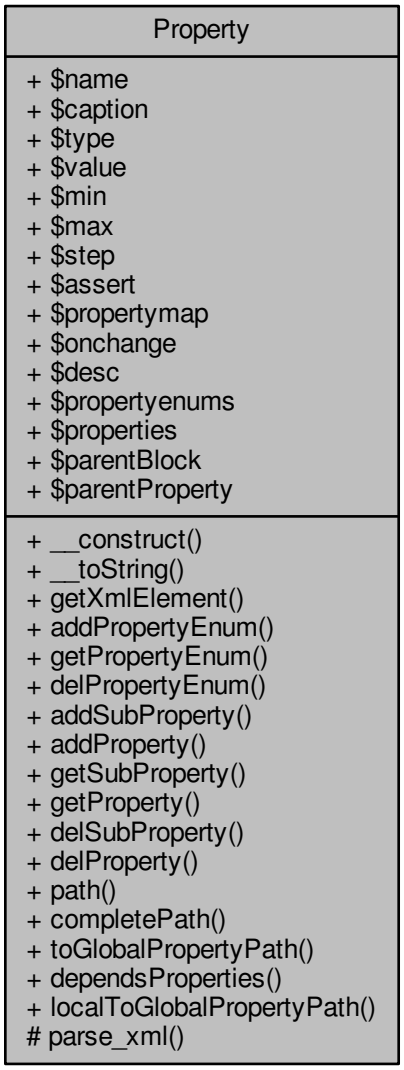
The documentation for this class was generated from the following file:

- model/process.php

6.37 Property Class Reference

The Property class is used to define high level properties.

Collaboration diagram for Property:



Public Member Functions

- __construct ($xml=null)

  constructor of Property
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance
- addPropertyEnum ($propertyenum)

  Add a property enum to the block.
- getPropertyEnum ($name, $casesens=true)

  return a reference to the property enum with the name $name, if not found, return false

- delPropertyEnum ($name)

  delete a property enum from his name
- addSubProperty ($property)

  Add a sub-property to the property.
- addProperty ($property)

  alias to addSubProperty($property)
- getSubProperty ($name, $casesens=true)

  return a reference to the property with the name $name, if not found, return false
- getProperty ($name, $casesens=true)

  alias to getSubProperty($name, $casesens)
- delSubProperty ($name)

  delete a property from his name
- delProperty ($name)

  alias to delSubProperty($name)
- path ()

  return the path of the property (without the name of the block)
- completePath ()

  Returns the complete path for the property with the name of the block.
- toGlobalPropertyPath ($blockName)

  Transforms the propertymap expression into a global expression in the context of $blockName.

Static Public Member Functions

- static dependsProperties ($expression)

  Returns all the depending property of the expression $expression.
- static localToGlobalPropertyPath ($expression, $blockName)

  Transforms an expression into a global expression in the context of $blockName.

Public Attributes

- $name

  Name of the property.
- $caption

  Caption of the property for high level interface.
- $type

  Type of property.
- $value

  Current value of the property.
- $min

  Minimum value of the property.
- $max

  Maximum value of the property.
- $step

  Step value of the property.
- $assert

  Check if the property is good (optional)
- $propertymap

  Mapping to properties (optional)
- $onchange

  Code to execute if the property change (optional)

- $desc

  Description of the param (optional)
- $propertyenums

  Array of enums if param property have different enums (optional)
- $properties

  Array of property class specify the high level properties.
- $parentBlock

  Reference to the associated parent block (don't set if parentProperty is set)
- $parentProperty

  Reference to the associated parent Property (don't set if parentBlock is set)

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.37.1 Detailed Description

The Property class is used to define high level properties.

Property is contained into the Block::$properties and Flow::$properties. It maps mathematical relation between hardware registers and software logical properties. When a property change, all dependent propreties are re-evaluated.

See Also

Block Param ParamBitfield Flow

6.37.2 Constructor & Destructor Documentation

6.37.2.1 __construct (*$xml* = `null`)

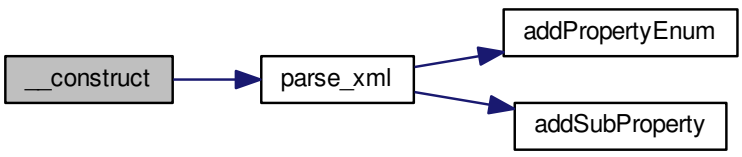
constructor of Property

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.37.3 Member Function Documentation

6.37.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

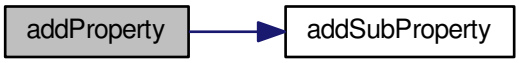
6.37.3.2 addProperty (*$property*)

alias to addSubProperty($property)

Parameters

Property	*$property*	sub-property to add to the property

Here is the call graph for this function:



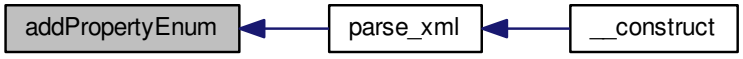
6.37.3.3 addPropertyEnum (*$propertyenum*)

Add a property enum to the block.

Parameters

Property-Enum	*$propertyenum*	property enum to add to the property

Here is the caller graph for this function:



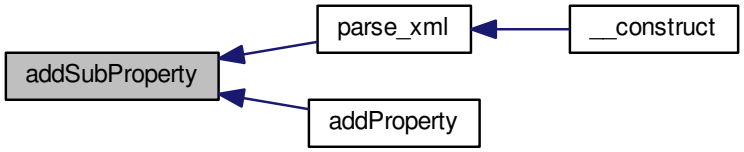
6.37.3.4 addSubProperty (*$property*)

Add a sub-property to the property.

Parameters

Property	*$property*	sub-property to add to the property

Here is the caller graph for this function:



6.37.3.5 completePath ( )

Returns the complete path for the property with the name of the block.

If the property is a subproperty of another property, the call is recursive.

Returns

string complete path of the property

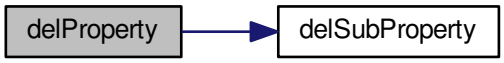
6.37.3.6 delProperty (*$name*)

alias to delSubProperty($name)

Parameters

string	*$name*	name of the property to delete

Here is the call graph for this function:



6.37.3.7 delPropertyEnum (*$name*)

delete a property enum from his name

Parameters

string	*$name*	name of the property enum to delete

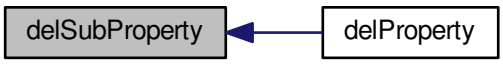
6.37.3.8 delSubProperty (*$name*)

delete a property from his name

Parameters

string	*$name*	name of the property to delete

Here is the caller graph for this function:



6.37.3.9 static dependsProperties (*$expression*) [static]

Returns all the depending property of the expression $expression.

Parameters

string	*$expression*	expression that need to be analysed

Returns

array|string array of property path

Here is the caller graph for this function:

6.37.3.10 getProperty (*$name, $casesens* = true)

alias to getSubProperty($name, $casesens)

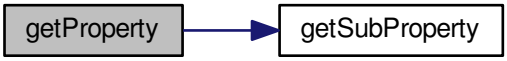
Parameters

string	*$name*	name of the property enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property

Here is the call graph for this function:



6.37.3.11 getPropertyEnum (*$name, $casesens* = `true`)

return a reference to the property enum with the name $name, if not found, return false

Parameters

string	*$name*	name of the property enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

PropertyEnum found property enum

6.37.3.12 getSubProperty (*$name, $casesens* = `true`)

return a reference to the property with the name $name, if not found, return false

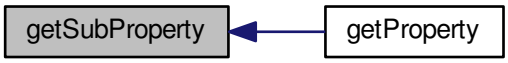
Parameters

string	*$name*	name of the property enum to search
bool	*$casesens*	take care or not of the case of the name

Returns

Property found property

Here is the caller graph for this function:



6.37.3.13 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.37.3.14 static localToGlobalPropertyPath (*$expression, $blockName*) [static]

Transforms an expression into a global expression in the context of $blockName.

This function use Property::dependsProperties function.

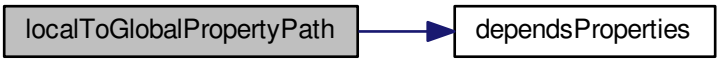
Parameters

string	*$expression*	expression to convert
string	*$blockName*	context of expression (name of the block)

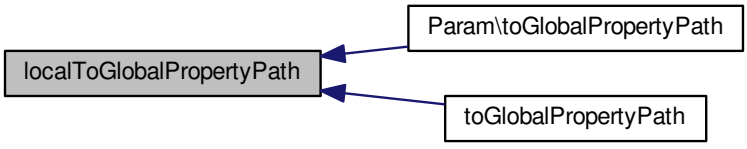
Returns

string processed expression in global context

Here is the call graph for this function:



Here is the caller graph for this function:



6.37.3.15 parse_xml (*$xml*) [protected]

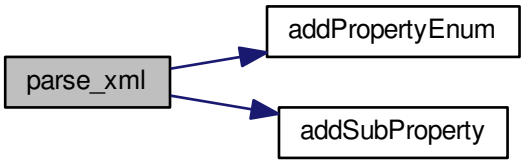
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.37.3.16 path ( )

return the path of the property (without the name of the block)

Returns

string path of the property

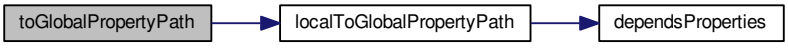
6.37.3.17 toGlobalPropertyPath (*$blockName*)

Transforms the propertymap expression into a global expression in the context of $blockName.

Parameters

string	*$blockName*	context of expression (name of the block)

Here is the call graph for this function:



6.37.4 Member Data Documentation

6.37.4.1 string $assert

Check if the property is good (optional)

6.37.4.2 string $caption

Caption of the property for high level interface.

6.37.4.3 string $desc

Description of the param (optional)

6.37.4.4 string $max

Maximum value of the property.

6.37.4.5 string $min

Minimum value of the property.

6.37.4.6 string $name

Name of the property.

6.37.4.7 string $onchange

Code to execute if the property change (optional)

6.37.4.8 Block $parentBlock

Reference to the associated parent block (don't set if parentProperty is set)

6.37.4.9 Property $parentProperty

Reference to the associated parent Property (don't set if parentBlock is set)

6.37.4.10 array Property $properties

Array of property class specify the high level properties.

6.37.4.11 array PropertyEnum $propertyenums

Array of enums if param property have different enums (optional)

6.37.4.12 string $propertymap

Mapping to properties (optional)

6.37.4.13 string $step

Step value of the property.

6.37.4.14 string $type

Type of property.

6.37.4.15 string $value

Current value of the property.

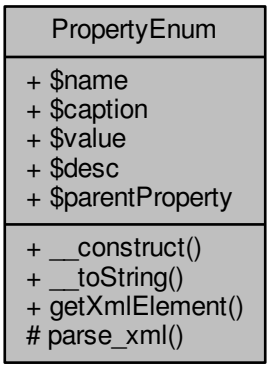
The documentation for this class was generated from the following file:

- model/property.php

6.38 PropertyEnum Class Reference

The PropertyEnum can be used to list the values that can take a property.

Collaboration diagram for PropertyEnum:



Public Member Functions

- __construct ($xml=null)

  constructor of PropertyEnum
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $name

  Name of the enum.
- $caption

  Caption of the property for high level interface (can contain space)
- $value

  Coresponding value of the enum when this enum is chosen.
- $desc

  Description of the enum (optional)
- $parentProperty

  Reference to the associated parent Property.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.38.1 Detailed Description

The PropertyEnum can be used to list the values that can take a property.

PropertyEnum is contained into the Property::$propertyenums when property have a list of choices for value.

See Also

Property

6.38.2 Constructor & Destructor Documentation

6.38.2.1 __construct (*$xml* = `null`)

constructor of PropertyEnum

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.38.3 Member Function Documentation

6.38.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.38.3.2 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document

string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.38.3.3 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.38.4 Member Data Documentation

6.38.4.1 string $caption

Caption of the property for high level interface (can contain space)

6.38.4.2 string $desc

Description of the enum (optional)

6.38.4.3 string $name

Name of the enum.

6.38.4.4 Property $parentProperty

Reference to the associated parent Property.

6.38.4.5 string $value

Coresponding value of the enum when this enum is chosen.

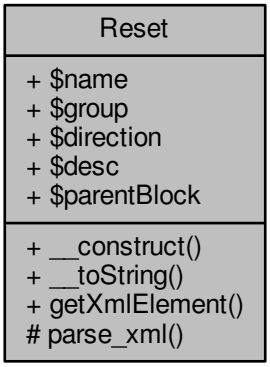
The documentation for this class was generated from the following file:

- model/propertyenum.php

6.39 Reset Class Reference

The Reset class define a reset input or reset provider in a Component or Block.

Collaboration diagram for Reset:



Public Member Functions

- __construct ($xml=null)

  constructor of Reset
- __toString ()

  funtion that export as string the main content of the class instance
- getXmlElement ($xml)

  permits to output this instance

Public Attributes

- $name

  Name of the reset.
- $group

  Name of the group reset.
- $direction

  Direction of the reset to specify if it's a source reset ('out') or an input reset ('in')
- $desc

  Description of the reset (optional)
- $parentBlock

  Reference to the associated parent block.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.39.1 Detailed Description

The Reset class define a reset input or reset provider in a Component or Block.

Reset is used into the Component::$resets to list all the reset of the block. A reset have information about the direction (in or out / provider or receiver), the group of reset and the name.

See Also

Component

6.39.2 Constructor & Destructor Documentation

6.39.2.1 __construct (*$xml* = `null`)

constructor of Reset

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.39.3 Member Function Documentation

6.39.3.1 __toString ( )

funtion that export as string the main content of the class instance

Returns

string

6.39.3.2 getXmlElement (*$xml*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.39.3.3 parse_xml (*$xml*) [protected]

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.39.4 Member Data Documentation

6.39.4.1 string $desc

Description of the reset (optional)

6.39.4.2 string $direction

Direction of the reset to specify if it's a source reset ('out') or an input reset ('in')

6.39.4.3 string $group

Name of the group reset.

6.39.4.4 string $name

Name of the reset.

6.39.4.5 Block $parentBlock

Reference to the associated parent block.

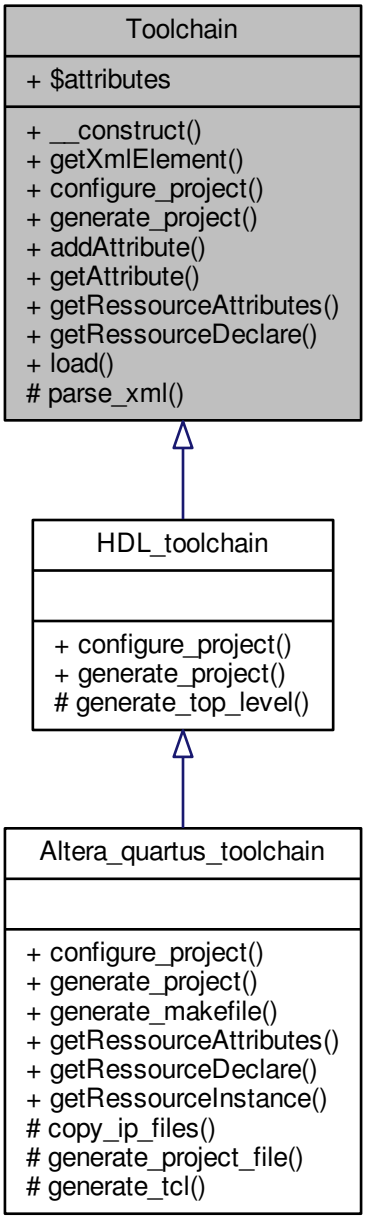
The documentation for this class was generated from the following file:

- model/reset.php

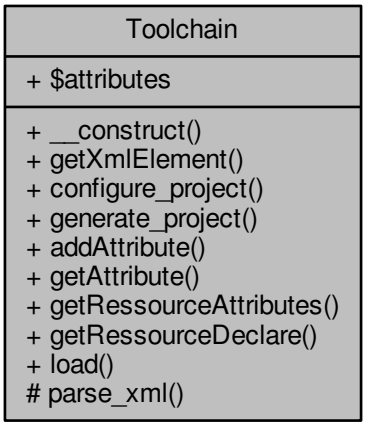
6.40 Toolchain Class Reference

Toolchain class define a toolchain for building a project.

Inheritance diagram for Toolchain:



Collaboration diagram for Toolchain:



Public Member Functions

- __construct ($xml=null)

  constructor of Toolchain
- getXmlElement ($xml, $format)

  permits to output this instance
- configure_project ($node)

  Configure the project node $node to be ready to be generated.
- generate_project ($node, $path)

  Create all necessary files to be compiled by the specified toolchain. Files generated depends of the used toolchain.
- addAttribute ($attribute)
- getAttribute ($name)
- getRessourceAttributes ($type)

  This function is used for special ressources attributes request.
- getRessourceDeclare ($type, $params)

  This function is used for special ressources assignement request.

Static Public Member Functions

- static load ($toolchain_name, $xml=null)

Public Attributes

- $attributes

  Array of attributes of the toolchain (optional)

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.40.1 Detailed Description

Toolchain class define a toolchain for building a project.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 __construct (*$xml* = `null`)

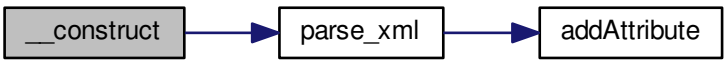
constructor of Toolchain

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.40.3 Member Function Documentation

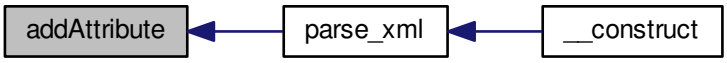
6.40.3.1 addAttribute (*$attribute*)

Add a attribute to the toolchain

Parameters

Attribute	*$attribute*	attribute to add to the toolchain

Here is the caller graph for this function:



6.40.3.2 configure_project (*$node*)

Configure the project node $node to be ready to be generated.

Parameters

Node	*$node*	node to configure

See Also

Toolchain::generate_project

6.40.3.3 generate_project (*$node, $path*)

Create all necessary files to be compiled by the specified toolchain. Files generated depends of the used toolchain.

Parameters

Node	*$node*	node structure
string	*$path*	output path of the generated project

6.40.3.4 getAttribute (*$name*)

return a reference to the attribute with the name $name, if not found, return false

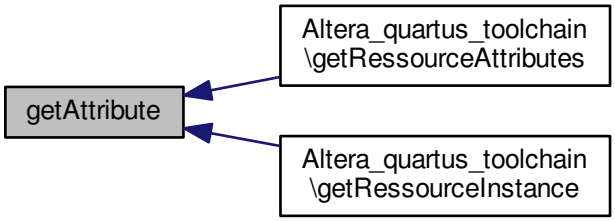
Parameters

string	*$name*	name of the attribute enum to search

Returns

Attribute found attribute

Here is the caller graph for this function:



6.40.3.5 getRessourceAttributes (*$type*)

This function is used for special ressources attributes request.

Parameters

string	*$type*	Type of ressource requested.

Returns

array

6.40.3.6 getRessourceDeclare (*$type, $params*)

This function is used for special ressources assignement request.

Parameters

string	*$type*	Type of ressource requested.

Returns

array

6.40.3.7 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.40.3.8 static load (*$toolchain_name, $xml* = `null`) `[static]`

6.40.3.9 parse_xml (*$xml*) `[protected]`

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:

Here is the caller graph for this function:

6.40.4 Member Data Documentation

6.40.4.1 array Attribute $attributes

Array of attributes of the toolchain (optional)

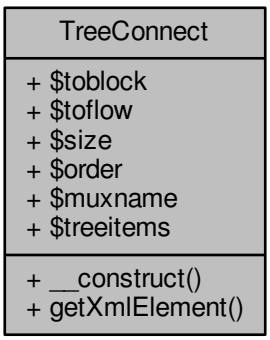
The documentation for this class was generated from the following file:

- model/toolchain.php

6.41 TreeConnect Class Reference

The TreeConnect class define a connection between two flows.

Collaboration diagram for TreeConnect:



Public Member Functions

- __construct ($toblock= ”, $toflow= ”, $size=8, $order= ’msb’, $muxname= ”)
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $toblock

  Name of the block sink of the flow.
- $toflow

  Name of the flow on the block sink of the flow.
- $size

  Size of connect in bit.
- $order

  Byte ordering can be "msb" or "lsb", default value is "msb".
- $muxname

  Property name of the MUX.
- $treeitems

  List of all the source flow can be chosen for this flow input.

6.41.1 Detailed Description

The TreeConnect class define a connection between two flows.

Reset is used into the Component::$resets to list all the reset of the block. Compared to FlowConnect, this interface is processed to obtain all the connection with the generated MUX.

See Also

PI Flow

6.41.2 Constructor & Destructor Documentation

6.41.2.1 __construct (*$toblock* = '', *$toflow* = '', *$size* = 8, *$order* = 'msb', *$muxname* = '')

6.41.3 Member Function Documentation

6.41.3.1 getXmlElement (*$xml*, *$format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.41.4 Member Data Documentation

6.41.4.1 string $muxname

Property name of the MUX.

6.41.4.2 string $order

Byte ordering can be "msb" or "lsb", default value is "msb".

6.41.4.3 string $size

Size of connect in bit.

6.41.4.4 string $toblock

Name of the block sink of the flow.

6.41.4.5 string $toflow

Name of the flow on the block sink of the flow.

6.41.4.6 array TreeItem $treeitems

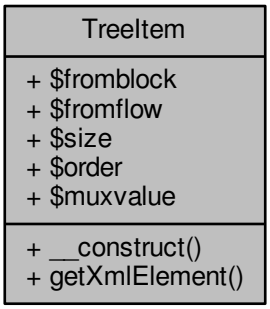
List of all the source flow can be chosen for this flow input.

The documentation for this class was generated from the following file:

- model/treeconnect.php

6.42 TreeItem Class Reference

Collaboration diagram for TreeItem:



Public Member Functions

- __construct ($fromblock= ”, $fromflow= ”, $size=8, $order= ’msb’, $muxvalue= ”)
- getXmlElement ($xml, $format)

  permits to output this instance

Public Attributes

- $fromblock

  Name of the block source of the flow.
- $fromflow

  Name of the flow on the block source of the flow.
- $size

  Size of connect in bit.
- $order

  Byte ordering can be "msb" or "lsb", default value is "msb".
- $muxvalue

  Value of the property to choose this connexion.

6.42.1 Constructor & Destructor Documentation

6.42.1.1 __construct (*$fromblock = ”, $fromflow = ”, $size =* `8`, *$order =* `'msb'`, *$muxvalue = ”*)

6.42.2 Member Function Documentation

6.42.2.1 getXmlElement (*$xml, $format*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document
string	*$format*	desired output file format

Returns

DOMElement xml element corresponding to this current instance

6.42.3 Member Data Documentation

6.42.3.1 string $fromblock

Name of the block source of the flow.

6.42.3.2 string $fromflow

Name of the flow on the block source of the flow.

6.42.3.3 string $muxvalue

Value of the property to choose this connexion.

6.42.3.4 string $order

Byte ordering can be "msb" or "lsb", default value is "msb".

6.42.3.5 string $size

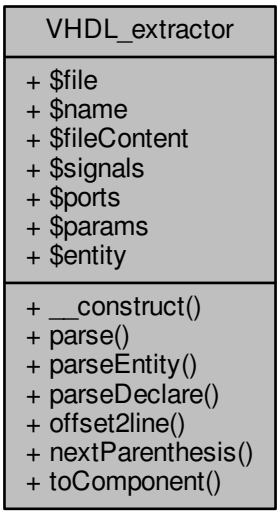
Size of connect in bit.

The documentation for this class was generated from the following file:

- model/treeitem.php

6.43 VHDL_extractor Class Reference

Collaboration diagram for VHDL_extractor:



Public Member Functions

- __construct ($file="")
- parse ($file)
- parseEntity ()
- parseDeclare ($part)
- offset2line ($offset)
- nextParenthesis ($offset)
- toComponent ($tool)

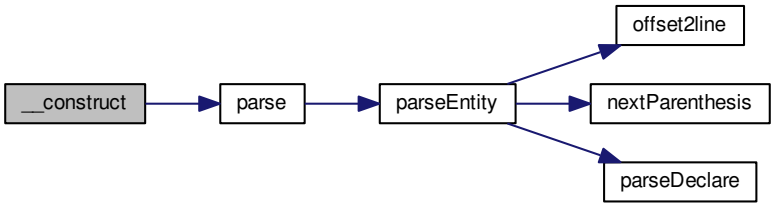
Public Attributes

- $file
- $name
- $fileContent
- $signals
- $ports
- $params
- $entity

6.43.1 Constructor & Destructor Documentation

6.43.1.1 __construct (*$file* = " ")

Here is the call graph for this function:



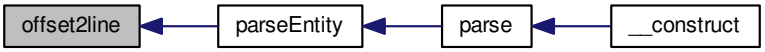
6.43.2 Member Function Documentation

6.43.2.1 nextParenthesis (*$offset*)

Here is the caller graph for this function:

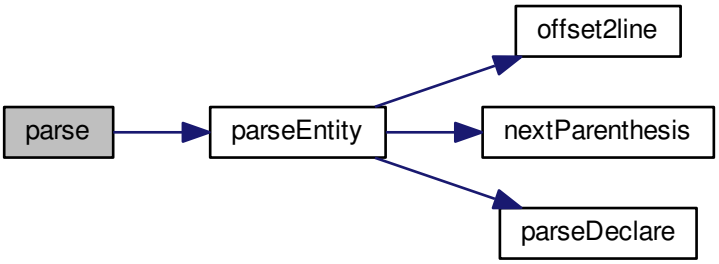
6.43.2.2 offset2line (*$offset*)

Here is the caller graph for this function:

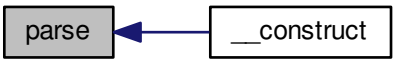


6.43.2.3 parse (*$file*)

Here is the call graph for this function:

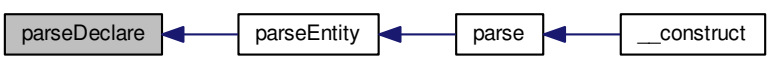


Here is the caller graph for this function:



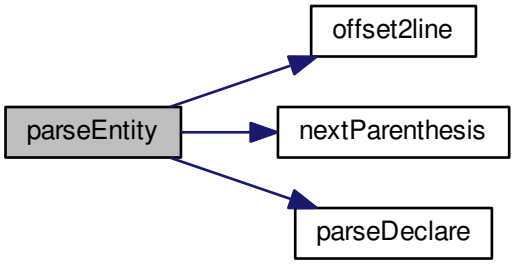
6.43.2.4 parseDeclare (*$part*)

Here is the caller graph for this function:

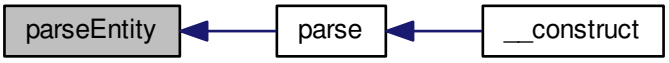


6.43.2.5 parseEntity ( )

Here is the call graph for this function:



Here is the caller graph for this function:



6.43.2.6 toComponent (*$tool*)

6.43.3 Member Data Documentation

6.43.3.1 $entity

6.43.3.2 $file

6.43.3.3 $fileContent

6.43.3.4 $name

6.43.3.5 $params

6.43.3.6 $ports

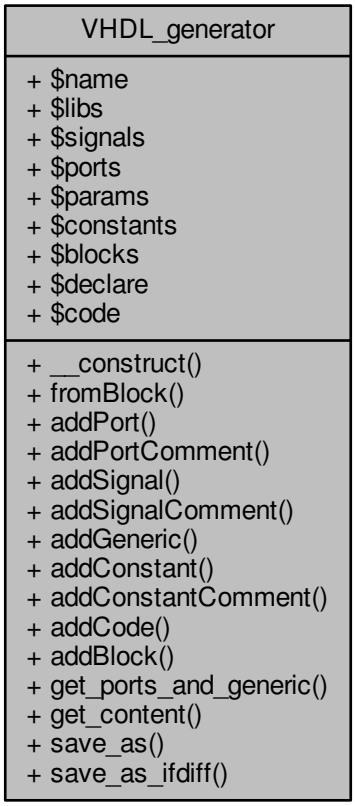
6.43.3.7 $signals

The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/vhdl_extractor.php

6.44 VHDL_generator Class Reference

Collaboration diagram for VHDL_generator:



Public Member Functions

- __construct ($name= 'module')
- fromBlock ($block, $subblock=FALSE)
- addPort ($name, $size, $direction, $type= '')
- addPortComment ($comment)
- addSignal ($name, $size, $type)
- addSignalComment ($comment)
- addGeneric ($name, $value)
- addConstant ($name, $value, $type)
- addConstantComment ($comment)
- addCode ($code)
- addBlock ($block, $subblock=FALSE)
- get_ports_and_generic ()
- get_content ()
- save_as ($filename)
- save_as_ifdiff ($filename)

Public Attributes

- $name
- $libs
- $signals
- $ports
- $params
- $constants
- $blocks
- $declare
- $code

6.44.1 Constructor & Destructor Documentation

6.44.1.1 __construct (*$name* = `'module'`)

6.44.2 Member Function Documentation

6.44.2.1 addBlock (*$block*, *$subblock* = `FALSE`)

Add a block to list of blocks

Parameters

Block	*$block*	block to add to the blocks
bool	*$subblock*	indicate if the block is a sub block or not

6.44.2.2 addCode (*$code*)

6.44.2.3 addConstant (*$name*, *$value*, *$type*)

6.44.2.4 addConstantComment (*$comment*)

6.44.2.5 addGeneric (*$name*, *$value*)

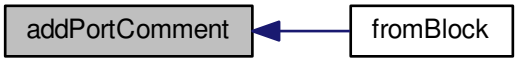
Here is the caller graph for this function:

6.44.2.6 addPort (*$name, $size, $direction, $type = ''*)

Here is the caller graph for this function:

6.44.2.7 addPortComment (*$comment*)

Here is the caller graph for this function:



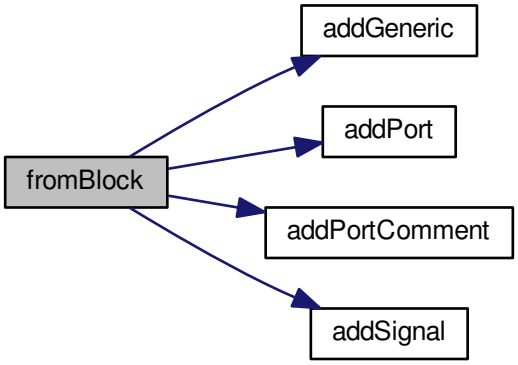
6.44.2.8 addSignal (*$name, $size, $type*)

Here is the caller graph for this function:

6.44.2.9 addSignalComment (*$comment*)

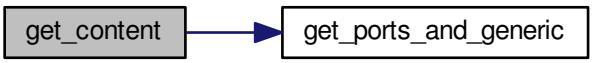
6.44.2.10 fromBlock (*$block*, *$subblock* = FALSE)

Here is the call graph for this function:

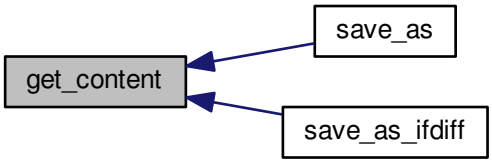


6.44.2.11 get_content ( )

Here is the call graph for this function:

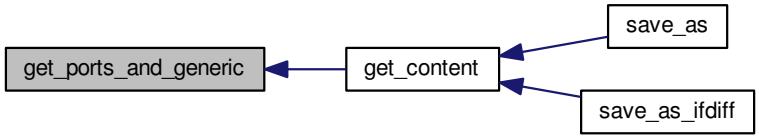


Here is the caller graph for this function:



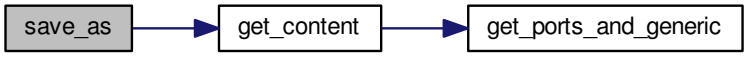
6.44.2.12 get_ports_and_generic ( )

Here is the caller graph for this function:



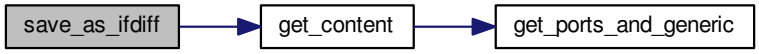
6.44.2.13 save_as (*$filename*)

Here is the call graph for this function:



6.44.2.14 save_as_ifdiff (*$filename*)

Here is the call graph for this function:



6.44.3 Member Data Documentation

6.44.3.1 $blocks

6.44.3.2 $code

6.44.3.3 $constants

6.44.3.4 $declare

6.44.3.5 $libs

6.44.3.6 $name

6.44.3.7 $params

6.44.3.8 $ports

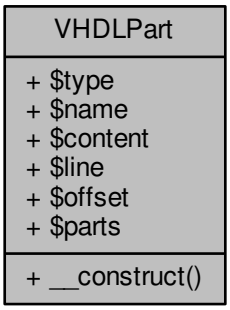
6.44.3.9 $signals

The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/vhdl_generator.php

6.45 VHDLPart Class Reference

Collaboration diagram for VHDLPart:



Public Member Functions

- __construct ($type, $name, $content, $line, $offset)

Public Attributes

- $type
- $name
- $content
- $line
- $offset
- $parts

6.45.1 Constructor & Destructor Documentation

6.45.1.1 __construct (*$type, $name, $content, $line, $offset*)

6.45.2 Member Data Documentation

6.45.2.1 $content

6.45.2.2 $line

6.45.2.3 $name

6.45.2.4 $offset

6.45.2.5 $parts

6.45.2.6 $type

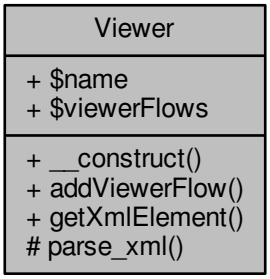
The documentation for this class was generated from the following file:

- /var/www/gpstudio/GPStudio_lib/support/toolchain/hdl/vhdl_extractor.php

6.46 Viewer Class Reference

The Viewer class define a viewer input or viewer provider.

Collaboration diagram for Viewer:



Public Member Functions

- __construct ($xml=null)

  constructor of Viewer
- addViewerFlow ($flow)

  Add a flow to the viewer.
- getXmlElement ($xml)

  permits to output this instance

Public Attributes

- $name

  Name of the viewer.
- $viewerFlows

  Array of ViewerFlow associated to this viewer.

Protected Member Functions

- parse_xml ($xml)

  internal function to fill this instance from input xml structure

6.46.1 Detailed Description

The Viewer class define a viewer input or viewer provider.

Viewer is used into the Block::$viewers to list all the viewer of the block.

See Also

Property

6.46.2 Constructor & Destructor Documentation

6.46.2.1 __construct (*$xml* = `null`)

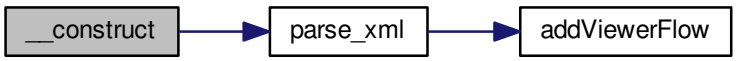
constructor of Viewer

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:



6.46.3 Member Function Documentation

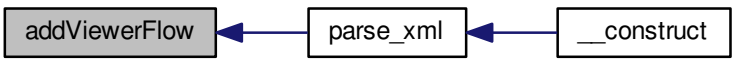
6.46.3.1 addViewerFlow (*$flow*)

Add a flow to the viewer.

Parameters

ViewerFlow	*$flow*	property to add to the block

Here is the caller graph for this function:



6.46.3.2 getXmlElement (*$xml*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOM-Document	*$xml*	reference of the output xml document

Returns

DOMElement xml element corresponding to this current instance

6.46.3.3 parse_xml (*$xml*) [protected]

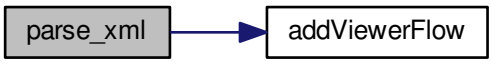
internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXM-LElement	*$xml*	xml element to parse

Here is the call graph for this function:



Here is the caller graph for this function:

6.46.4 Member Data Documentation

6.46.4.1 string $name

Name of the viewer.

6.46.4.2 array ViewerFlow $viewerFlows

Array of ViewerFlow associated to this viewer.

See Also

ViewerFlow

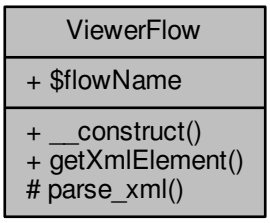
The documentation for this class was generated from the following file:

- model/viewer.php

6.47 ViewerFlow Class Reference

The ViewerFlow class define a flow connection to a viewer.

Collaboration diagram for ViewerFlow:



Public Member Functions

- __construct ($xml=null)
  constructor of ViewerFlow
- getXmlElement ($xml)
  permits to output this instance

Public Attributes

- $flowName
  Name of the associated flow.

Protected Member Functions

- parse_xml ($xml)
  internal function to fill this instance from input xml structure

6.47.1 Detailed Description

The ViewerFlow class define a flow connection to a viewer.

ViewerFlow is used into the Viewer::$viewerflows to list associated flow on each viewer.

See Also

Viewer

6.47.2 Constructor & Destructor Documentation

6.47.2.1 __construct (*$xml* = `null`)

constructor of ViewerFlow

Initialise all the internal members and call parse_xml if $xml is set

Parameters

SimpleXMLElement | null	*$xml*	if it's different of null, call the xml parser to fill members

Here is the call graph for this function:

6.47.3 Member Function Documentation

6.47.3.1 getXmlElement (*$xml*)

permits to output this instance

Return a formated node for the node_generated file. This method call all the children getXmlElement to add into this node.

Parameters

DOMDocument	*$xml*	reference of the output xml document

Returns

DOMElement xml element corresponding to this current instance

6.47.3.2 parse_xml (*$xml*) `[protected]`

internal function to fill this instance from input xml structure

Can be call only from this node into the constructor

Parameters

SimpleXMLElement	*$xml*	xml element to parse

Here is the caller graph for this function:

6.47.4 Member Data Documentation

6.47.4.1 string $flowName

Name of the associated flow.

The documentation for this class was generated from the following file:

- model/viewerflow.php

Index

$addr_abs
Block, 28
$addr_bus_width
ParamInterconnect, 168
$assert
Property, 191
$attributes
Component, 88
Pin, 171
Toolchain, 203
$bitfield
ParamBitfield, 164
$bitfieldlist
ParamBitfield, 164
$block
Block_generator, 31
$block_generator
Block_generator, 31
$blockname
InterfaceBus, 131
$blocks
Node, 154
VHDL_generator, 215
$board
Node, 154
$board_file
Board, 35
$boards
Lib, 144
$canbechain
PLL, 173
$caption
Property, 191
PropertyEnum, 194
$categ
Component, 89
$clkByPLL
PLL, 173
$clock_providers
ClockInterconnect, 54
$clock_receivers
ClockInterconnect, 54
$clocks
Board, 35
Component, 89
$code
VHDL_generator, 215
$comConnects
ComDriver, 62
$comParams
ComDriver, 62
$comdriver
IOCom, 141
$components
Component, 89
Lib, 145
$configscriptfile
Block, 28
Board, 35
$constants
VHDL_generator, 215
$content
VHDLPart, 216
$declare
VHDL_generator, 215
$default
Module_param, 147
Param, 159
$desc
Block, 28
Clock, 43
Component, 89
File, 104
Flow, 110
Param, 159
ParamBitfield, 164
Port, 177
Property, 191
PropertyEnum, 194
Reset, 197
$direction
Clock, 43
Reset, 197
$divmax
PLL, 173
$domain
Clock, 43
$domains
ClockInterconnect, 54
$driver
Component, 89
$driverio
ComDriver, 62
$dummyClc
PLL, 173
$entity
VHDL_extractor, 210
$ext_ports
Component, 89
$file
VHDL_extractor, 210
$fileContent
VHDL_extractor, 210
$filePath
LibItem, 146
$files
Component, 89
$flowName
ViewerFlow, 222
$flow_connects

FlowInterconnect, 120
$flows
Component, 89
$fromblock
FlowConnect, 113
TreeItem, 206
$fromflow
FlowConnect, 113
TreeItem, 206
$generated
File, 104
$generatescriptfile
Block, 28
Board, 36
$group
File, 104
Reset, 197
$hard
Param, 159
$height
ComponentPartProperty, 101
$id
ComConnect, 57
$inFreqs
PLL, 173
$in_lib
Component, 89
$infos
Component, 89
$ios
Lib, 145
$libs
VHDL_generator, 215
$line
VHDLPart, 216
$link
ComConnect, 57
$mapto
Pin, 171
$master_count
Block, 28
$max
Clock, 43
Param, 159
Property, 191
$min
Clock, 43
Param, 160
Property, 191
$mul
PLL, 173
$mulmax
PLL, 173
$muxname
TreeConnect, 204
$muxvalue
TreeItem, 206
$name
Attribute, 16
Board, 36
Clock, 43
ClockDomain, 46
ComParam, 66
Component, 89
ComponentPart, 95
ComponentPartFlow, 98
ComponentPartProperty, 101
File, 104
Flow, 110
Info, 129
InterfaceBus, 131
LibItem, 146
Module_param, 147
Node, 155
Param, 160
ParamBitfield, 164
Pin, 171
Port, 177
Property, 191
PropertyEnum, 194
Reset, 197
VHDL_extractor, 210
VHDL_generator, 215
VHDLPart, 216
Viewer, 219
$net
Clock, 43
$node_file
Node, 155
$offset
VHDLPart, 216
$onchange
Property, 191
$order
FlowConnect, 113
TreeConnect, 204
TreeItem, 206
$outFreqs
PLL, 173
$parambitfields
Param, 160
$params
Component, 89
VHDL_extractor, 210
VHDL_generator, 215
$parentBlock
Clock, 44
File, 104
Flow, 110
Param, 160
Pin, 171
Port, 177
Property, 191
Reset, 197
$parentComponent
Component, 89

$parentNode
Block, 28
Board, 36
$parentParam
ParamBitfield, 164
$parentProperty
Property, 191
PropertyEnum, 194
$partflows
ComponentPart, 95
$partproperties
ComponentPart, 95
$parts
Component, 89
VHDLPart, 217
$path
Board, 36
Component, 89
File, 104
$pi_size_addr_rel
Block, 29
$pins
Board, 36
IO, 136
$plls
ClockInterconnect, 54
$ports
VHDL_extractor, 210
VHDL_generator, 215
$process_generator
Block_generator, 32
$processes
Lib, 145
$properties
Block, 29
Flow, 110
Property, 191
$propertyenums
Property, 191
$propertymap
Param, 160
ParamBitfield, 164
Property, 191
$ratio
Clock, 44
$regaddr
Param, 160
$resets
Board, 36
Component, 90
$shift
Clock, 44
$signals
VHDL_extractor, 210
VHDL_generator, 216
$size
Flow, 110
Module_param, 147
Port, 177
TreeConnect, 204
TreeItem, 206
$size_addr
InterfaceBus, 131
$slave_generator
Block_generator, 32
$space
Module_param, 147
$step
Property, 191
$svg
ComponentPart, 95
$toblock
FlowConnect, 113
TreeConnect, 204
$toflow
FlowConnect, 113
TreeConnect, 204
$toolchain
Board, 36
Lib, 145
$tree_connects
FlowInterconnect, 120
$treeitems
TreeConnect, 204
$type
Attribute, 16
ComConnect, 57
File, 104
Flow, 110
InterfaceBus, 131
Module_param, 147
Param, 160
ParamBitfield, 164
Port, 177
Property, 191
VHDLPart, 217
$typical
Clock, 44
ClockDomain, 46
$value
Attribute, 16
ComParam, 66
Info, 129
Param, 160
ParamBitfield, 164
Property, 191
PropertyEnum, 194
$vco
PLL, 173
$vcomax
PLL, 174
$vcomin
PLL, 174
$viewerFlows
Viewer, 219
$viewers

GPViewer, 124
$virtual_pll
ClockInterconnect, 54
$width
ComponentPartProperty, 101
$x_pos
ComponentPart, 95
ComponentPartFlow, 98
ComponentPartProperty, 101
$xml
Component, 90
Node, 155
$y_pos
ComponentPart, 95
ComponentPartFlow, 98
ComponentPartProperty, 101
__construct
Attribute, 14
Block, 20
Block_generator, 30
Board, 33
Clock, 38
ClockDomain, 45
ClockInterconnect, 50
ComConnect, 56
ComDriver, 59
ComParam, 64
Component, 71
ComponentPart, 91
ComponentPartFlow, 97
ComponentPartProperty, 100
File, 103
Flow, 106
FlowConnect, 112
FlowInterconnect, 116
GPViewer, 122
Info, 128
InterfaceBus, 130
IO, 134
IOCom, 140
Lib, 143
LibItem, 146
Module_param, 147
Node, 149
Param, 157
ParamBitfield, 162
ParamInterconnect, 167
Pin, 170
PLL, 173
Port, 175
Process, 180
Property, 184
PropertyEnum, 193
Reset, 196
Toolchain, 200
TreeConnect, 204
TreeItem, 205
VHDL_extractor, 207
VHDL_generator, 212
VHDLPart, 216
Viewer, 218
ViewerFlow, 221
__toString
Clock, 39
File, 103
Flow, 106
Info, 129
Param, 157
ParamBitfield, 162
Port, 175
Property, 185
PropertyEnum, 193
Reset, 196

addAttribute
Component, 72
Toolchain, 200
addBlock
Node, 149
VHDL_generator, 212
addClock
Board, 34
Component, 73
addClockDomain
ClockInterconnect, 50
addCode
VHDL_generator, 212
addComConnect
ComDriver, 59
addComParam
ComDriver, 59
addComponent
Component, 73
addConstant
VHDL_generator, 212
addConstantComment
VHDL_generator, 212
addExtPort
Component, 73
addFile
Component, 74
addFlow
Component, 74
addFlowConnect
FlowInterconnect, 116
addFreq
PLL, 173
addGeneric
VHDL_generator, 212
addInfo
Component, 75
addInterface
Component, 75
addIo
Board, 34
Node, 149
addParam

Component, 75
addParamBitfield
Param, 157
addPart
Component, 76
addPartFlow
ComponentPart, 92
addPartProperty
ComponentPart, 92
addPin
Board, 34
IO, 135
addPort
VHDL_generator, 212
addPortComment
VHDL_generator, 213
addProcess
Node, 150
addProperty
Block, 20
Flow, 106
Property, 185
addPropertyEnum
Property, 185
addReset
Board, 34
Component, 76
addSignal
VHDL_generator, 213
addSignalComment
VHDL_generator, 213
addSubProperty
Property, 185
addViewer
GPViewer, 123
addViewerFlow
Viewer, 218
Altera_quartus_toolchain, 8
configure_project, 10
copy_ip_files, 10
generate_makefile, 10
generate_project, 10
generate_project_file, 11
generate_tcl, 11
getRessourceAttributes, 11
getRessourceDeclare, 12
getRessourceInstance, 12
Attribute, 12
$name, 16
$type, 16
$value, 16
__construct, 14
getXmlElement, 15
parse_xml, 15
availableIosName
Board, 34

Base script model, 6
bin
Block_generator, 30
Block, 16
$addr_abs, 28
$configscriptfile, 28
$desc, 28
$generatescriptfile, 28
$master_count, 28
$parentNode, 28
$pi_size_addr_rel, 29
$properties, 29
__construct, 20
addProperty, 20
configure, 21
delParamBitField, 21
delProperty, 21
delPropertyEnumPath, 22
delPropertyPath, 22
generate, 22
getParamBitField, 22
getProperty, 23
getPropertyEnumPath, 23
getPropertyPath, 24
getSubProperty, 24
getXmlElement, 26
parse_xml, 27
print_flow, 27
saveBlockDef, 27
toGlobalPropertyPath, 28
type, 28
Block_generator, 29
$block, 31
$block_generator, 31
$process_generator, 32
$slave_generator, 32
__construct, 30
bin, 30
convertData2Img, 30
convertImg2stim, 30
fromBlock, 30
generateProcess, 31
generateSlave, 31
generateTb, 31
generateTopBlock, 31
hex, 31
Board, 32
$board_file, 35
$clocks, 35
$configscriptfile, 35
$generatescriptfile, 36
$name, 36
$parentNode, 36
$path, 36
$pins, 36
$resets, 36
$toolchain, 36
__construct, 33
addClock, 34
addIo, 34

addPin, 34
addReset, 34
availableIosName, 34
configure, 34
generate, 34
getClock, 34
getPin, 34
getReset, 35
getXmlElement, 35
redefParam, 35

canBeAdded
PLL, 173
checkBlock
Lib, 143
checklib
Lib, 143
Clock, 36
$desc, 43
$direction, 43
$domain, 43
$max, 43
$min, 43
$name, 43
$net, 43
$parentBlock, 44
$ratio, 44
$shift, 44
$typical, 44
__construct, 38
__toString, 39
convert, 39
formatFreq, 40
getXmlElement, 40
hdlFreq, 42
parse_xml, 42
ClockDomain, 44
$name, 46
$typical, 46
__construct, 45
getXmlElement, 46
parse_xml, 46
ClockInterconnect, 47
$clock_providers, 54
$clock_receivers, 54
$domains, 54
$plls, 54
$virtual_pll, 54
__construct, 50
addClockDomain, 50
configure, 51
create_sdc, 51
generate, 52
getClockDomain, 52
getXmlElement, 53
parse_xml, 53
ppcm, 53
ppcm_array, 53
providerByFreq, 54
type, 54
cmp_raddr
Param, 157
ComConnect, 55
$id, 57
$link, 57
$type, 57
__construct, 56
getXmlElement, 56
parse_xml, 56
ComDriver, 57
$comConnects, 62
$comParams, 62
$driverio, 62
__construct, 59
addComConnect, 59
addComParam, 59
getComConnect, 61
getComParam, 61
getXmlElement, 61
parse_xml, 61
ComParam, 63
$name, 66
$value, 66
__construct, 64
getXmlElement, 65
parse_xml, 65
completePath
Property, 186
Component, 66
$attributes, 88
$categ, 89
$clocks, 89
$components, 89
$desc, 89
$driver, 89
$ext_ports, 89
$files, 89
$flows, 89
$in_lib, 89
$infos, 89
$name, 89
$params, 89
$parentComponent, 89
$parts, 89
$path, 89
$resets, 90
$xml, 90
__construct, 71
addAttribute, 72
addClock, 73
addComponent, 73
addExtPort, 73
addFile, 74
addFlow, 74
addInfo, 75
addInterface, 75
addParam, 75

addPart, 76
addReset, 76
componentsList, 77
delClock, 77
delComponent, 77
delExtPort, 77
delFileByPath, 77
delFlow, 78
delParam, 78
delReset, 78
getAttribute, 78
getClock, 78
getComponent, 79
getExtPort, 79
getFile, 79
getFileByPath, 81
getFlow, 81
getInfo, 82
getInstance, 82
getInterface, 83
getParam, 83
getPart, 84
getReset, 84
getXmlElement, 85
parse_xml, 85
saveComponentDef, 86
setSvgDraw, 88
type, 88
ComponentPart, 90
$name, 95
$partflows, 95
$partproperties, 95
$svg, 95
$x_pos, 95
$y_pos, 95
__construct, 91
addPartFlow, 92
addPartProperty, 92
delPartFlow, 92
delPartProperty, 94
getPartFlow, 94
getPartProperty, 94
getXmlElement, 94
parse_xml, 94
ComponentPartFlow, 96
$name, 98
$x_pos, 98
$y_pos, 98
__construct, 97
getXmlElement, 97
parse_xml, 98
ComponentPartProperty, 98
$height, 101
$name, 101
$width, 101
$x_pos, 101
$y_pos, 101
__construct, 100
getXmlElement, 100
parse_xml, 100
componentsList
Component, 77
configure
Block, 21
Board, 34
ClockInterconnect, 51
FlowInterconnect, 117
ParamInterconnect, 167
configure_project
Altera_quartus_toolchain, 10
HDL_toolchain, 126
Toolchain, 200
convert
Clock, 39
convertData2Img
Block_generator, 30
convertImg2stim
Block_generator, 30
copy_ip_files
Altera_quartus_toolchain, 10
create_dot_file
FlowInterconnect, 117
create_header_file
ParamInterconnect, 167
create_sdc
ClockInterconnect, 51

decodeBitField
ParamBitfield, 162
delClock
Component, 77
delComponent
Component, 77
delExtPort
Component, 77
delFileByPath
Component, 77
delFlow
Component, 78
delFlowConnect
FlowInterconnect, 118
delFlowConnectToBlock
FlowInterconnect, 118
delIo
Node, 150
delParam
Component, 78
delParamBitField
Block, 21
delParambitfield
Param, 158
delPartFlow
ComponentPart, 92
delPartProperty
ComponentPart, 94
delProcess
Node, 150

delProperty
Block, 21
Flow, 108
Property, 186
delPropertyEnum
Property, 186
delPropertyEnumPath
Block, 22
delPropertyPath
Block, 22
delReset
Component, 78
delSubProperty
Property, 186
dependsProperties
Property, 187

File, 101
$desc, 104
$generated, 104
$group, 104
$name, 104
$parentBlock, 104
$path, 104
$type, 104
__construct, 103
__toString, 103
getXmlElement, 103
parse_xml, 104
Flow, 105
$desc, 110
$name, 110
$parentBlock, 110
$properties, 110
$size, 110
$type, 110
__construct, 106
__toString, 106
addProperty, 106
delProperty, 108
getProperty, 108
getSubProperty, 108
getXmlElement, 109
parse_xml, 109
FlowConnect, 110
$fromblock, 113
$fromflow, 113
$order, 113
$toblock, 113
$toflow, 113
__construct, 112
getXmlElement, 112
parse_xml, 112
FlowInterconnect, 113
$flow_connects, 120
$tree_connects, 120
__construct, 116
addFlowConnect, 116
configure, 117
create_dot_file, 117
delFlowConnect, 118
delFlowConnectToBlock, 118
generate, 118
getFlowConnect, 118
getXmlElement, 119
parse_xml, 119
showBlocks, 119
showConnect, 119
showFlows, 120
type, 120
formatFreq
Clock, 40
freqIn
PLL, 173
fromBlock
Block_generator, 30
VHDL_generator, 213

GPViewer, 120
$viewers, 124
__construct, 122
addViewer, 123
getViewer, 123
getXmlElement, 123
parse_xml, 124
generate
Block, 22
Board, 34
ClockInterconnect, 52
FlowInterconnect, 118
ParamInterconnect, 168
generate_makefile
Altera_quartus_toolchain, 10
generate_project
Altera_quartus_toolchain, 10
HDL_toolchain, 126
Toolchain, 201
generate_project_file
Altera_quartus_toolchain, 11
generate_tcl
Altera_quartus_toolchain, 11
generate_top_level
HDL_toolchain, 127
generateProcess
Block_generator, 31
generateSlave
Block_generator, 31
generateTb
Block_generator, 31
generateTopBlock
Block_generator, 31
get_content
VHDL_generator, 214
get_ports_and_generic
VHDL_generator, 214
getAttribute
Component, 78
Toolchain, 201

getBlock
Node, 150
getClock
Board, 34
Component, 78
getClockDomain
ClockInterconnect, 52
getComConnect
ComDriver, 61
getComParam
ComDriver, 61
getComponent
Component, 79
getExtPort
Component, 79
getFile
Component, 79
getFileByPath
Component, 81
getFlow
Component, 81
Node, 151
getFlowConnect
FlowInterconnect, 118
getInfo
Component, 82
getInstance
Component, 82
getInterface
Component, 83
getParam
Component, 83
getParamBitField
Block, 22
getParambitfield
Param, 158
getPart
Component, 84
getPartFlow
ComponentPart, 94
getPartProperty
ComponentPart, 94
getPin
Board, 34
IO, 135
getProperty
Block, 23
Flow, 108
Property, 187
getPropertyEnum
Property, 188
getPropertyEnumPath
Block, 23
getPropertyPath
Block, 24
getReset
Board, 35
Component, 84
getRessourceAttributes
Altera_quartus_toolchain, 11
Toolchain, 201
getRessourceDeclare
Altera_quartus_toolchain, 12
Toolchain, 201
getRessourceInstance
Altera_quartus_toolchain, 12
getSubProperty
Block, 24
Flow, 108
Property, 188
getViewer
GPViewer, 123
getXmlElement
Attribute, 15
Block, 26
Board, 35
Clock, 40
ClockDomain, 46
ClockInterconnect, 53
ComConnect, 56
ComDriver, 61
ComParam, 65
Component, 85
ComponentPart, 94
ComponentPartFlow, 97
ComponentPartProperty, 100
File, 103
Flow, 109
FlowConnect, 112
FlowInterconnect, 119
GPViewer, 123
Info, 129
IO, 135
IOCom, 140
Node, 151
Param, 158
ParamBitfield, 163
ParamInterconnect, 168
Pin, 170
Port, 175
Process, 181
Property, 188
PropertyEnum, 193
Reset, 196
Toolchain, 202
TreeConnect, 204
TreeItem, 205
Viewer, 218
ViewerFlow, 221

HDL_toolchain, 125
configure_project, 126
generate_project, 126
generate_top_level, 127
hdlFreq
Clock, 42
hex

Block_generator, 31
ParamInterconnect, 168

IO, 131
$pins, 136
__construct, 134
addPin, 135
getPin, 135
getXmlElement, 135
parse_xml, 136
type, 136
IOCom, 137
$comdriver, 141
__construct, 140
getXmlElement, 140
parse_xml, 141
type, 141
Info, 127
$name, 129
$value, 129
__construct, 128
__toString, 129
getXmlElement, 129
parse_xml, 129
InterfaceBus, 130
$blockname, 131
$name, 131
$size_addr, 131
$type, 131
__construct, 130
isempty
PLL, 173

Lib, 141
$boards, 144
$components, 145
$ios, 145
$processes, 145
$toolchain, 145
__construct, 143
checkBlock, 143
checklib, 143
listboard, 144
listcomponent, 144
listio, 144
listprocess, 144
listtoolchain, 144
openlib, 144
LibItem, 145
$filePath, 146
$name, 146
__construct, 146
listboard
Lib, 144
listcomponent
Lib, 144
listio
Lib, 144
listprocess
Lib, 144
listtoolchain
Lib, 144
load
Toolchain, 202
localToGlobalPropertyPath
Property, 189

Module_param, 146
$default, 147
$name, 147
$size, 147
$space, 147
$type, 147
__construct, 147

nextParenthesis
VHDL_extractor, 208
Node, 147
$blocks, 154
$board, 154
$name, 155
$node_file, 155
$xml, 155
__construct, 149
addBlock, 149
addIo, 149
addProcess, 150
delIo, 150
delProcess, 150
getBlock, 150
getFlow, 151
getXmlElement, 151
parse_config_xml, 152
saveProject, 152
saveXml, 154
setBoard, 154

offset2line
VHDL_extractor, 208
openlib
Lib, 144

PLL, 171
$canbechain, 173
$clkByPLL, 173
$divmax, 173
$dummyClc, 173
$inFreqs, 173
$mul, 173
$mulmax, 173
$outFreqs, 173
$vco, 173
$vcomax, 174
$vcomin, 174
__construct, 173
addFreq, 173
canBeAdded, 173
freqIn, 173

isempty, 173
Param, 155
$default, 159
$desc, 159
$hard, 159
$max, 159
$min, 160
$name, 160
$parambitfields, 160
$parentBlock, 160
$propertymap, 160
$regaddr, 160
$type, 160
$value, 160
__construct, 157
__toString, 157
addParamBitfield, 157
cmp_raddr, 157
delParambitfield, 158
getParambitfield, 158
getXmlElement, 158
parse_xml, 158
toGlobalPropertyPath, 159
ParamBitfield, 160
$bitfield, 164
$bitfieldlist, 164
$desc, 164
$name, 164
$parentParam, 164
$propertymap, 164
$type, 164
$value, 164
__construct, 162
__toString, 162
decodeBitField, 162
getXmlElement, 163
parse_xml, 163
ParamInterconnect, 164
$addr_bus_width, 168
__construct, 167
configure, 167
create_header_file, 167
generate, 168
getXmlElement, 168
hex, 168
type, 168
parse
VHDL_extractor, 208
parse_config_xml
Node, 152
parse_xml
Attribute, 15
Block, 27
Clock, 42
ClockDomain, 46
ClockInterconnect, 53
ComConnect, 56
ComDriver, 61
ComParam, 65
Component, 85
ComponentPart, 94
ComponentPartFlow, 98
ComponentPartProperty, 100
File, 104
Flow, 109
FlowConnect, 112
FlowInterconnect, 119
GPViewer, 124
Info, 129
IO, 136
IOCom, 141
Param, 158
ParamBitfield, 163
Pin, 171
Port, 177
Process, 181
Property, 189
PropertyEnum, 194
Reset, 197
Toolchain, 202
Viewer, 219
ViewerFlow, 221
parseDeclare
VHDL_extractor, 209
parseEntity
VHDL_extractor, 209
path
Property, 190
Pin, 168
$attributes, 171
$mapto, 171
$name, 171
$parentBlock, 171
__construct, 170
getXmlElement, 170
parse_xml, 171
Port, 174
$desc, 177
$name, 177
$parentBlock, 177
$size, 177
$type, 177
__construct, 175
__toString, 175
getXmlElement, 175
parse_xml, 177
ppcm
ClockInterconnect, 53
ppcm_array
ClockInterconnect, 53
print_flow
Block, 27
Process, 178
__construct, 180
getXmlElement, 181
parse_xml, 181

type, 181
Property, 181
$assert, 191
$caption, 191
$desc, 191
$max, 191
$min, 191
$name, 191
$onchange, 191
$parentBlock, 191
$parentProperty, 191
$properties, 191
$propertyenums, 191
$propertymap, 191
$step, 191
$type, 191
$value, 191
__construct, 184
__toString, 185
addProperty, 185
addPropertyEnum, 185
addSubProperty, 185
completePath, 186
delProperty, 186
delPropertyEnum, 186
delSubProperty, 186
dependsProperties, 187
getProperty, 187
getPropertyEnum, 188
getSubProperty, 188
getXmlElement, 188
localToGlobalPropertyPath, 189
parse_xml, 189
path, 190
toGlobalPropertyPath, 190
PropertyEnum, 192
$caption, 194
$desc, 194
$name, 194
$parentProperty, 194
$value, 194
__construct, 193
__toString, 193
getXmlElement, 193
parse_xml, 194
providerByFreq
ClockInterconnect, 54

redefParam
Board, 35
Reset, 194
$desc, 197
$direction, 197
$group, 197
$name, 197
$parentBlock, 197
__construct, 196
__toString, 196
getXmlElement, 196
parse_xml, 197

save_as
VHDL_generator, 215
save_as_ifdiff
VHDL_generator, 215
saveBlockDef
Block, 27
saveComponentDef
Component, 86
saveProject
Node, 152
saveXml
Node, 154
setBoard
Node, 154
setSvgDraw
Component, 88
showBlocks
FlowInterconnect, 119
showConnect
FlowInterconnect, 119
showFlows
FlowInterconnect, 120

toComponent
VHDL_extractor, 210
toGlobalPropertyPath
Block, 28
Param, 159
Property, 190
Toolchain, 197
$attributes, 203
__construct, 200
addAttribute, 200
configure_project, 200
generate_project, 201
getAttribute, 201
getRessourceAttributes, 201
getRessourceDeclare, 201
getXmlElement, 202
load, 202
parse_xml, 202
TreeConnect, 203
$muxname, 204
$order, 204
$size, 204
$toblock, 204
$toflow, 204
$treeitems, 204
__construct, 204
getXmlElement, 204
TreeItem, 205
$fromblock, 206
$fromflow, 206
$muxvalue, 206
$order, 206
$size, 206
__construct, 205

getXmlElement, 205
type
Block, 28
ClockInterconnect, 54
Component, 88
FlowInterconnect, 120
IO, 136
IOCom, 141
ParamInterconnect, 168
Process, 181

VHDL_extractor, 207
$entity, 210
$file, 210
$fileContent, 210
$name, 210
$params, 210
$ports, 210
$signals, 210
__construct, 207
nextParenthesis, 208
offset2line, 208
parse, 208
parseDeclare, 209
parseEntity, 209
toComponent, 210
VHDL_generator, 211
$blocks, 215
$code, 215
$constants, 215
$declare, 215
$libs, 215
$name, 215
$params, 215
$ports, 215
$signals, 216
__construct, 212
addBlock, 212
addCode, 212
addConstant, 212
addConstantComment, 212
addGeneric, 212
addPort, 212
addPortComment, 213
addSignal, 213
addSignalComment, 213
fromBlock, 213
get_content, 214
get_ports_and_generic, 214
save_as, 215
save_as_ifdiff, 215
VHDLPart, 216
$content, 216
$line, 216
$name, 216
$offset, 216
$parts, 217
$type, 217
__construct, 216

Viewer, 217
$name, 219
$viewerFlows, 219
__construct, 218
addViewerFlow, 218
getXmlElement, 218
parse_xml, 219
ViewerFlow, 220
$flowName, 222
__construct, 221
getXmlElement, 221
parse_xml, 221