Introduction to Altera IP Cores

2016.02.05

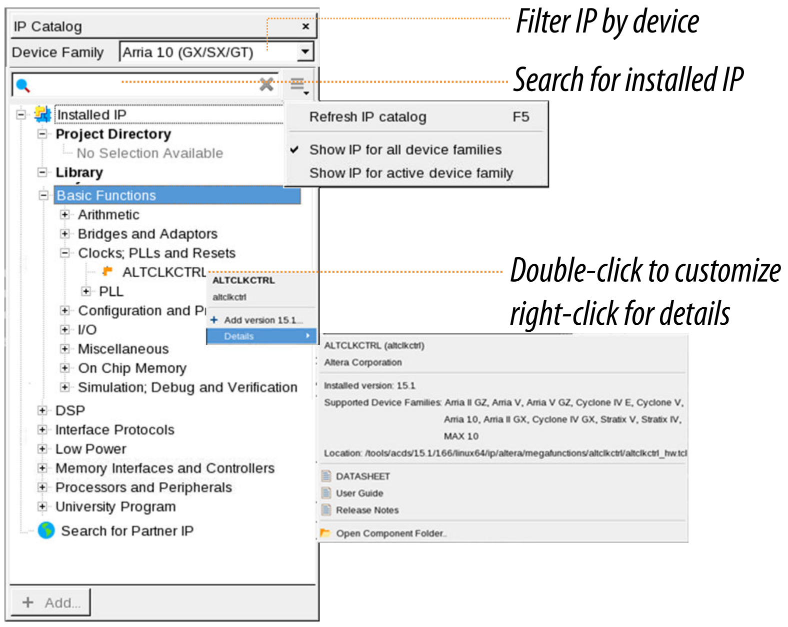
UG-01056 Subscribe Send Feedback

Altera® and strategic IP partners offer a broad portfolio of configurable IP cores optimized for Altera devices. The Quartus® Prime software installation includes the Altera IP library. You can integrate optimized and verified Altera IP cores into your design to shorten design cycles and maximize performance. The Quartus Prime software also supports integration of IP cores from other sources. Use the IP Catalog to efficiently parameterize and generate synthesis and simulation files for a custom IP variation. The Altera IP library includes the following types of IP cores:

- Basic functions
- DSP functions
- Interface protocols
- Low power functions
- Memory interfaces and controllers
- Processors and peripherals

IP Catalog and Parameter Editor

The IP Catalog (**Tools** > **IP Catalog**) and parameter editor help you easily customize and integrate IP cores into your project. Use the IP Catalog and parameter editor to select, customize, and generate files representing the custom IP variation in your project.



ALTCLKCTRL (altclkctrl)
Altera Corporation

Installed version: 15.1
Supported Device Families: Arria II GZ, Arria V, Arria V GZ, Cyclone IV E, Cyclone V, Arria 10, Arria II GX, Cyclone IV GX, Stratix V, Stratix IV, MAX 10
Location: /tools/acds/15.1/166/linux64/ip/altera/megafunctions/altclkctrl/altclkctrl_hw.tcl

© 2016 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, ENPIRION, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

ISO 9001:2008 Registered

The IP Catalog displays the installed IP cores available for your design. Double-click any IP core to launch the parameter editor and generate files representing your IP variation. Use the following features to help you quickly locate and select an IP core:

- Filter IP Catalog to **Show IP for active device family** or **Show IP for all device families**. If you have no project open, select the **Device Family** in IP Catalog.
- Type in the Search field to locate any full or partial IP core name in IP Catalog.
- Right-click an IP core name in IP Catalog to display details about supported devices, open the IP core's installation folder, and click links to IP documentation.
- Click **Search for Partner IP**, to access partner IP information on the Altera website.

The parameter editor prompts you to specify an IP variation name, optional ports, and output file generation options. The parameter editor generates a top-level Qsys system file (**.qsys**) or Quartus Prime IP file (**.qip**) representing the IP core in your project. You can also parameterize an IP variation without an open project.

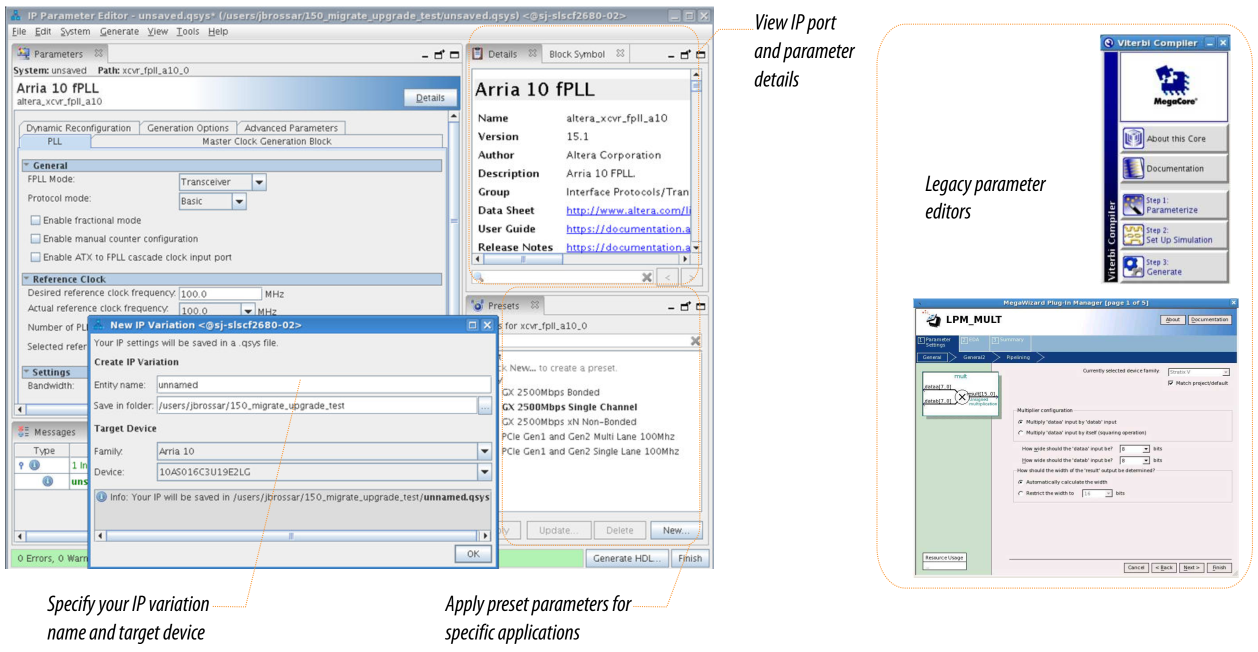
The IP Catalog is also available in Qsys (**View** > **IP Catalog**). The Qsys IP Catalog includes exclusive system interconnect, video and image processing, and other system-level IP that are not available in the Quartus Prime IP Catalog. For more information about using the Qsys IP Catalog, refer to *Creating a System with Qsys* in the *Quartus Prime Handbook*.

Note: The IP Catalog (**Tools** > **IP Catalog)** and parameter editor replace the MegaWizard™ Plug-In Manager for IP selection and parameterization, beginning in Quartus II software version 14.0. Use the IP Catalog and parameter editor to locate and paramaterize Altera IP cores.

Using the Parameter Editor

The parameter editor helps you to configure IP core ports, parameters, and output file generation options.

Figure 1: IP Parameter Editors


View IP port and parameter details
Legacy parameter editors
Specify your IP variation name and target device
Apply preset parameters for specific applications
File Edit System Generate View Tools Help
Parameters
Details
Block Symbol
Presets
Messages
Viterbi Compiler
About this Core
Documentation
Step 1: Parameterize
Step 2: Set Up Simulation
Step 3: Generate
LPM_MULT
Generate HDL...
Finish
OK

IP Parameter Editor - unsaved.qsys* (/users/jbrossar/150_migrate_upgrade_test/unsaved.qsys) <@sj-slscf2680-02>

System: unsaved Path: xcvr_fpll_a10_0

Arria 10 fPLL
altera_xcvr_fpll_a10

Dynamic Reconfiguration | Generation Options | Advanced Parameters
PLL | Master Clock Generation Block

General
FPLL Mode: Transceiver
Protocol mode: Basic
Enable fractional mode
Enable manual counter configuration
Enable ATX to FPLL cascade clock input port

Reference Clock
Desired reference clock frequency: 100.0 MHz
Actual reference clock frequency: 100.0 MHz

Settings
Bandwidth:

Name	altera_xcvr_fpll_a10
Version	15.1
Author	Altera Corporation
Description	Arria 10 FPLL
Group	Interface Protocols/Tran
Data Sheet	http://www.altera.com/li
User Guide	https://documentation.a
Release Notes	https://documentation.a

New IP Variation <@sj-slscf2680-02>
Your IP settings will be saved in a .qsys file.
Create IP Variation
Entity name: unnamed
Save in folder: /users/jbrossar/150_migrate_upgrade_test
Target Device
Family: Arria 10
Device: 10AS016C3U19E2LG
Info: Your IP will be saved in /users/jbrossar/150_migrate_upgrade_test/unnamed.qsys

Presets for xcvr_fpll_a10_0
New... to create a preset.
GX 2500Mbps Bonded
GX 2500Mbps Single Channel
GX 2500Mbps xN Non-Bonded
PCIe Gen1 and Gen2 Multi Lane 100MHz
PCIe Gen1 and Gen2 Single Lane 100Mhz
Update... Delete New...

0 Errors, 0 Warn

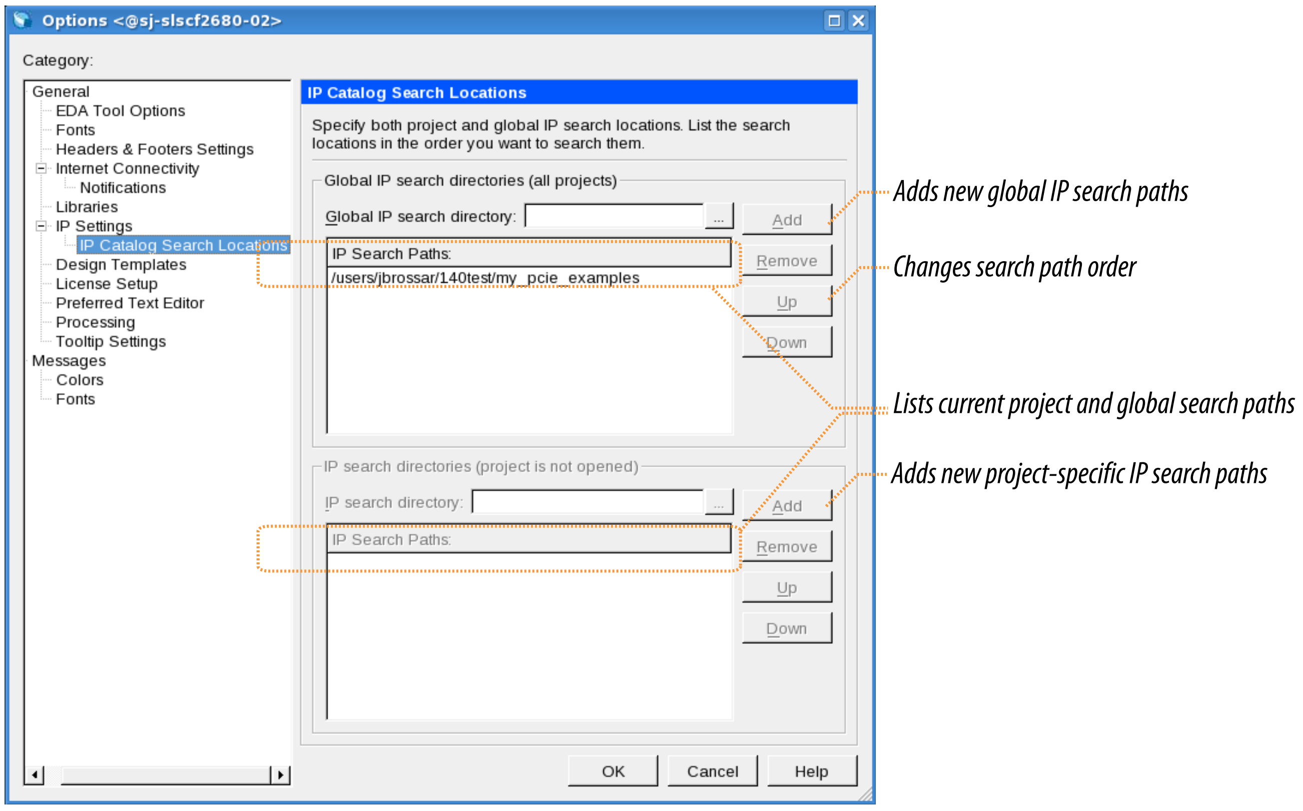
- Use preset settings in the parameter editor (where provided) to instantly apply preset parameter values for specific applications.
- View port and parameter descriptions, and click links to documentation.
- Generate testbench systems or example designs (where provided).

Adding IP Cores to IP Catalog

The IP Catalog automatically displays Altera IP cores found in the project directory, in the Altera installation directory, and in the defined IP search path. The IP Catalog can include Altera-provided IP components, third-party IP components, custom IP components that you provide, and previously generated Qsys systems.

You can use the **IP Search Path** option (**Tools** > **Options**) to include custom and third-party IP components in the IP Catalog. The IP Catalog displays all IP cores in the IP search path.

Figure 2: Specifying IP Search Locations


Options <@sj-slscf2680-02>
Category:
General
EDA Tool Options
Fonts
Headers & Footers Settings
Internet Connectivity
Notifications
Libraries
IP Settings
IP Catalog Search Locations
Design Templates
License Setup
Preferred Text Editor
Processing
Tooltip Settings
Messages
Colors
Fonts
IP Catalog Search Locations
Global IP search directories (all projects)
Global IP search directory:
Add
Remove
Up
Down
IP search directories (project is not opened)
IP search directory:
Add
Remove
Up
Down
OK
Cancel
Help
Adds new global IP search paths
Changes search path order
Lists current project and global search paths
Adds new project-specific IP search paths

Specify both project and global IP search locations. List the search locations in the order you want to search them.

IP Search Paths:
/users/jbrossar/140test/my_pcie_examples

IP Search Paths:

The Quartus Prime software searches the directories listed in the IP search path for the following IP core files:

- Component Description File (**_hw.tcl**)—Defines a single IP core.
- IP Index File (**.ipx**)—Each **.ipx** file indexes a collection of available IP cores, or a reference to other directories to search. In general, **.ipx** files facilitate faster searches.

The Quartus Prime software searches some directories recursively and other directories only to a specific depth. When the search is recursive, the search stops at any directory that contains an **_hw.tcl** or **.ipx** file.

In the following list of search locations, a recursive descent is annotated by **. A single * signifies any file.

Table 1: IP Search Locations

Location	Description
PROJECT_DIR/*	Finds IP components and index files in the Quartus Prime project directory.
PROJECT_DIR/ip//***	Finds IP components and index files in any subdirectory of the **/ip** subdirectory of the Quartus Prime project directory.

If the Quartus Prime software recognizes two IP cores with the same name, the following search path precedence rules determine the resolution of files:

1. Project directory.
2. Project database directory.
3. Project IP search path specified in **IP Search Locations**, or with the `SEARCH_PATH` assignment for the current project revision.
4. Global IP search path specified in **IP Search Locations**, or with the `SEARCH_PATH` assignment in the **quartus2.ini** file.
5. Quartus software libraries directory, such as **<*Quartus Installation*>\libraries**.

Note: If you add a component to the search path, you must update the IP Catalog by clicking **Refresh IP Catalog** in the drop-down list. In Qsys, click **File** > **Refresh System** to update the IP Catalog.

General Settings for IP

You can use the following settings to control how the Quartus Prime software manages IP cores in your project.

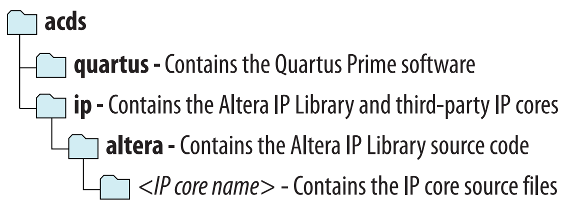
Table 2: IP Core General Setting Locations

Setting Location	Description
Tools > **Options** > **IP Settings** Or **Assignments** > **Settings** > **IP Settings** (only enabled with open project)	• Specify your **IP generation HDL preference**. The parameter editor generates IP files in your preferred HDL by default. • Increase **Maximum Qsys memory usage size** if you experience slow processing for large systems, or if Qsys reports an Out of Memory error. • Specify whether to **Automatically add Quartus Prime IP files** to all projects. Disable this option to control addition of IP files manually. You may want to experiment with IP before adding to a project. • Use the **IP Regeneration Policy** setting to control when synthesis files regenerate for each IP variation. Typically you **Always regenerate synthesis files for IP cores** after making changes to an IP variation.
Tools > **Options** > **IP Catalog Search Locations** Or **Assignments** > **Settings** > **IP Catalog Search Locations**	• Specify project and global IP search locations. The Quartus Prime software searches for IP cores in the project directory, in the Altera installation directory, and in the IP search path.

Licensing IP Cores

The Altera IP Library provides many useful IP core functions for your production use without purchasing an additional license. Some Altera MegaCore® IP functions require that you purchase a separate license for production use. However, the OpenCore® feature allows evaluation of any Altera IP core in simulation and compilation in the Quartus Prime software. After you are satisfied with functionality and performance, visit the Self Service Licensing Center to obtain a license number for any Altera product.

Figure 3: IP Core Installation Path


acds
quartus - Contains the Quartus Prime software
ip - Contains the Altera IP Library and third-party IP cores
altera - Contains the Altera IP Library source code
<IP core name> - Contains the IP core source files

Note: The default IP installation directory on Windows is **<*drive*>:\altera\***<version number>*; on Linux it is *<home directory>***/altera/** *<version number>*.

OpenCore Plus IP Evaluation

Altera's free OpenCore Plus feature allows you to evaluate licensed MegaCore IP cores in simulation and hardware before purchase. You need only purchase a license for MegaCore IP cores if you decide to take your design to production. OpenCore Plus supports the following evaluations:

- Simulate the behavior of a licensed IP core in your system.
- Verify the functionality, size, and speed of the IP core quickly and easily.
- Generate time-limited device programming files for designs that include IP cores.
- Program a device with your IP core and verify your design in hardware.

OpenCore Plus evaluation supports the following two operation modes:

- Untethered—run the design containing the licensed IP for a limited time.
- Tethered—run the design containing the licensed IP for a longer time or indefinitely. This requires a connection between your board and the host computer.

Note: All IP cores that use OpenCore Plus time out simultaneously when any IP core in the design times out.

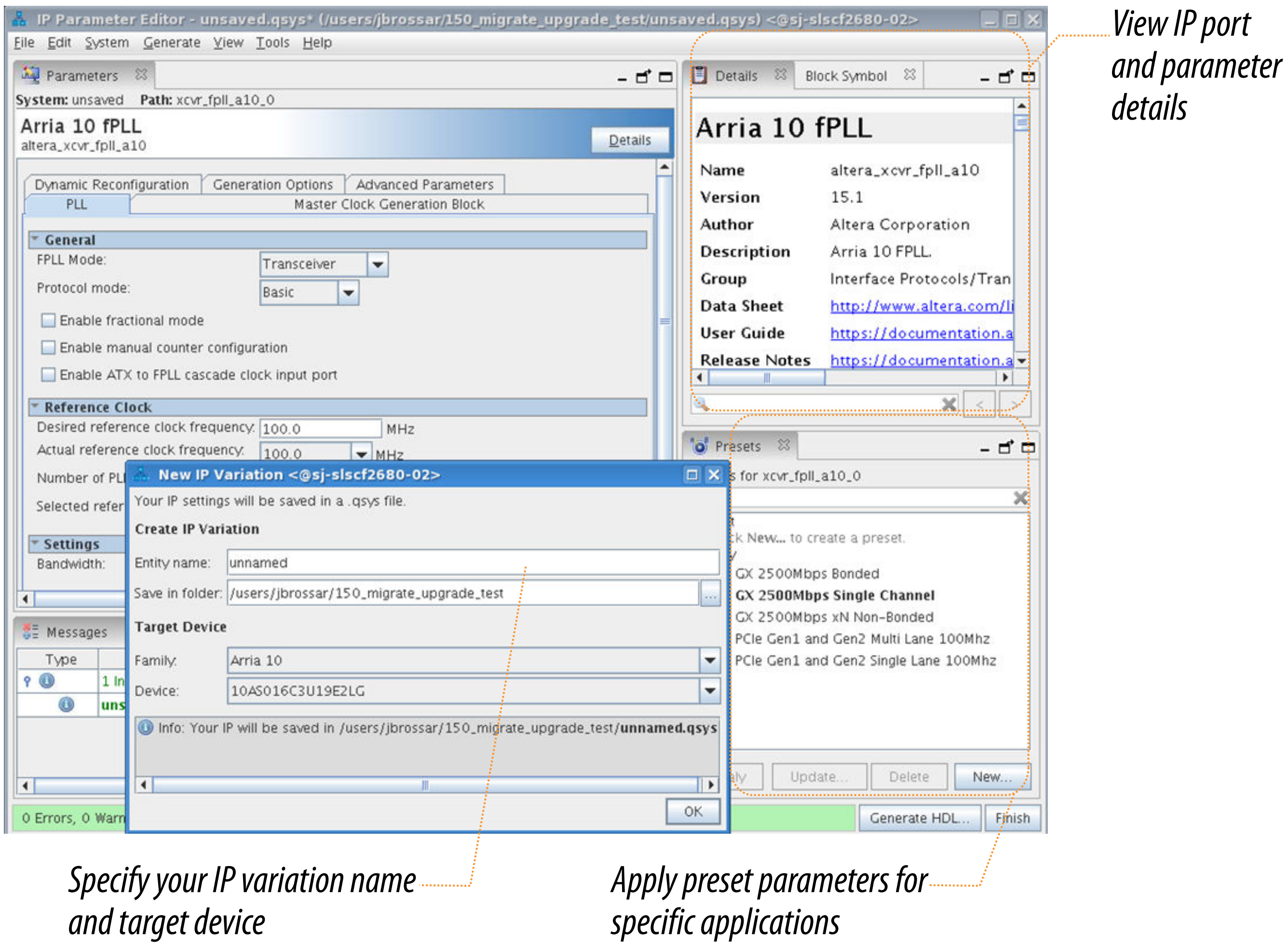
Related Information

- Altera Licensing Site
- Altera Software Installation and Licensing Manual

Generating IP Cores

You can quickly configure a custom IP variation in the parameter editor. Use the following steps to specify IP core options and parameters in the parameter editor.

Figure 4: IP Parameter Editor


IP Parameter Editor - unsaved.qsys* (/users/jbrossar/150_migrate_upgrade_test/unsaved.qsys) <@sj-slscf2680-02>
File Edit System Generate View Tools Help
Parameters
Details
Block Symbol
Dynamic Reconfiguration
Generation Options
Advanced Parameters
PLL
Master Clock Generation Block
Presets
Messages
Update...
Delete
New...
OK
Generate HDL...
Finish
View IP port and parameter details
Specify your IP variation name and target device
Apply preset parameters for specific applications

System: unsaved Path: xcvr_fpll_a10_0

Arria 10 fPLL
altera_xcvr_fpll_a10

General
FPLL Mode: Transceiver
Protocol mode: Basic
Enable fractional mode
Enable manual counter configuration
Enable ATX to FPLL cascade clock input port

Reference Clock
Desired reference clock frequency: 100.0 MHz
Actual reference clock frequency: 100.0 MHz
Number of PLL
Selected refer

Settings
Bandwidth:

Arria 10 fPLL

Name	altera_xcvr_fpll_a10
Version	15.1
Author	Altera Corporation
Description	Arria 10 FPLL.
Group	Interface Protocols/Tran
Data Sheet	http://www.altera.com/li
User Guide	https://documentation.a
Release Notes	https://documentation.a

s for xcvr_fpll_a10_0
k New... to create a preset.
GX 2500Mbps Bonded
GX 2500Mbps Single Channel
GX 2500Mbps xN Non-Bonded
PCIe Gen1 and Gen2 Multi Lane 100Mhz
PCIe Gen1 and Gen2 Single Lane 100Mhz

New IP Variation <@sj-slscf2680-02>
Your IP settings will be saved in a .qsys file.
Create IP Variation
Entity name: unnamed
Save in folder: /users/jbrossar/150_migrate_upgrade_test
Target Device
Family: Arria 10
Device: 10AS016C3U19E2LG
Info: Your IP will be saved in /users/jbrossar/150_migrate_upgrade_test/**unnamed.qsys**

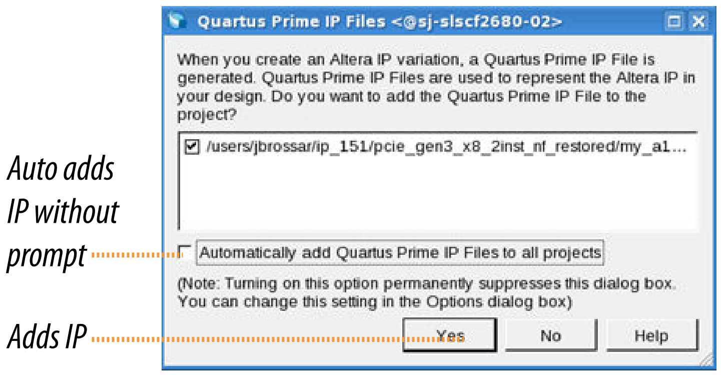
0 Errors, 0 Warn

1. In the IP Catalog (**Tools** > **IP Catalog**), locate and double-click the name of the IP core to customize. The parameter editor appears.
2. Specify a top-level name for your custom IP variation. The parameter editor saves the IP variation settings in a file named *<your_ip>*.**qsys**. Click **OK**. Do not include spaces in IP variation names or paths.
3. Specify the parameters and options for your IP variation in the parameter editor, including one or more of the following. Refer to your IP core user guide for information about specific IP core parameters.

- Optionally select preset parameter values if provided for your IP core. Presets specify initial parameter values for specific applications.
- Specify parameters defining the IP core functionality, port configurations, and device-specific features.
- Specify options for processing the IP core files in other EDA tools.

4. Click **Generate HDL**. The **Generation** dialog box appears.
5. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.
6. To generate a simulation testbench, click **Generate** > **Generate Testbench System**.
7. To generate an HDL instantiation template that you can copy and paste into your text editor, click **Generate** > **HDL Example**.
8. Click **Finish**. Click **Yes** if prompted to add files representing the IP variation to your project. Optionally turn on the option to **Automatically add Quartus Prime IP Files to All Projects**. Click **Project** > **Add/Remove Files in Project** to add IP files at any time.

Figure 5: Adding IP Files to Project


Auto adds IP without prompt
Adds IP
Quartus Prime IP Files <@sj-slscf2680-02>
Yes
No
Help

When you create an Altera IP variation, a Quartus Prime IP File is generated. Quartus Prime IP Files are used to represent the Altera IP in your design. Do you want to add the Quartus Prime IP File to the project?

☑ /users/jbrossar/ip_151/pcie_gen3_x8_2inst_nf_restored/my_a1...

☐ Automatically add Quartus Prime IP Files to all projects

(Note: Turning on this option permanently suppresses this dialog box. You can change this setting in the Options dialog box)

The generated **.qsys** file must be added to your project to represent IP and Qsys systems.

9. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

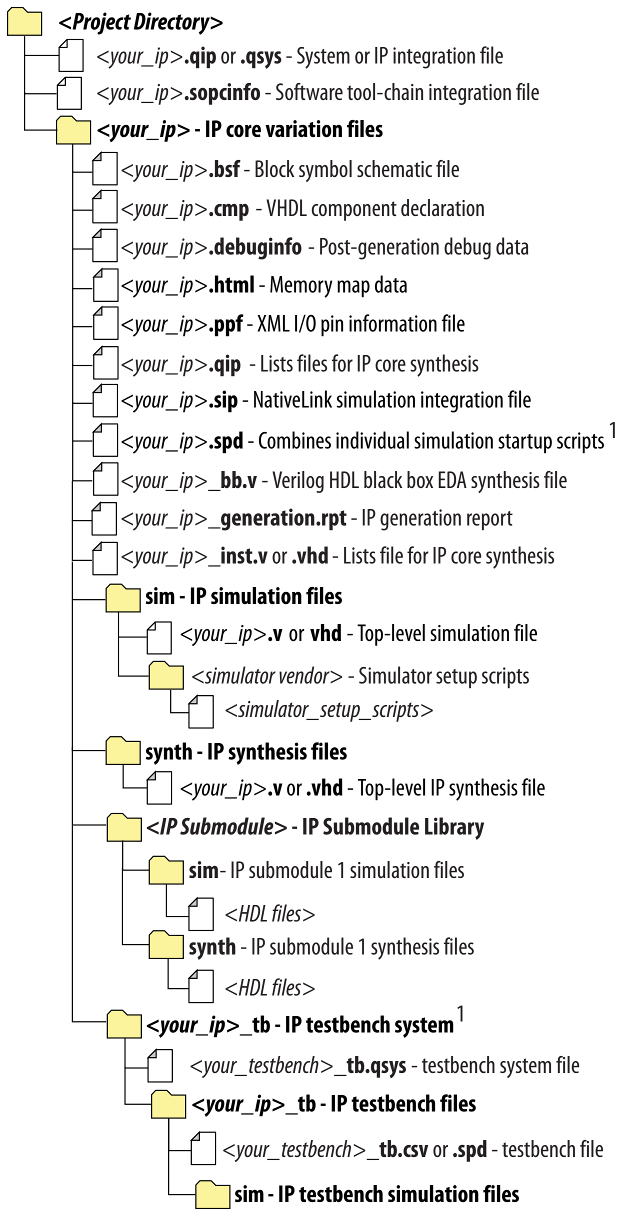
Related Information

- IP User Guide Documentation
- Altera IP Release Notes

Files Generated for Altera IP Cores and Qsys Systems

The Quartus Prime software generates the following output file structure for IP cores and Qsys systems. The generated **.qsys** file must be added to your project to represent IP and Qsys systems.

Figure 6: Files generated for IP cores and Qsys Systems


<Project Directory>
<your_ip>.qip or .qsys - System or IP integration file
<your_ip>.sopcinfo - Software tool-chain integration file
<your_ip> - IP core variation files
<your_ip>.bsf - Block symbol schematic file
<your_ip>.cmp - VHDL component declaration
<your_ip>.debuginfo - Post-generation debug data
<your_ip>.html - Memory map data
<your_ip>.ppf - XML I/O pin information file
<your_ip>.qip - Lists files for IP core synthesis
<your_ip>.sip - NativeLink simulation integration file
<your_ip>.spd - Combines individual simulation startup scripts 1
<your_ip>_bb.v - Verilog HDL black box EDA synthesis file
<your_ip>_generation.rpt - IP generation report
<your_ip>_inst.v or .vhd - Lists file for IP core synthesis
sim - IP simulation files
<your_ip>.v or vhd - Top-level simulation file
<simulator vendor> - Simulator setup scripts
<simulator_setup_scripts>
synth - IP synthesis files
<your_ip>.v or .vhd - Top-level IP synthesis file
<IP Submodule> - IP Submodule Library
sim- IP submodule 1 simulation files
<HDL files>
synth - IP submodule 1 synthesis files
<HDL files>
<your_ip>_tb - IP testbench system 1
<your_testbench>_tb.qsys - testbench system file
<your_ip>_tb - IP testbench files
<your_testbench>_tb.csv or .spd - testbench file
sim - IP testbench simulation files

1. If supported and enabled for your IP core variation.

Table 3: IP Core and Qsys Simulation Generated Files

File Name	Description
***<my_ip>*.qsys**	The Qsys system or top-level IP variation file. *<my_ip>* is the name that you give your IP variation.

File Name	Description
<*system*>.sopcinfo	Describes the connections and IP component parameterizations in your Qsys system. You can parse its contents to get requirements when you develop software drivers for IP components. Downstream tools such as the Nios II tool chain use this file. The **.sopcinfo** file and the **system.h** file generated for the Nios II tool chain include address map information for each slave relative to each master that accesses the slave. Different masters may have a different address map to access a particular slave component.
<*my_ip*>.cmp	The VHDL Component Declaration (**.cmp**) file is a text file that contains local generic and port definitions that you can use in VHDL design files.
<*my_ip*>.html	A report that contains connection information, a memory map showing the address of each slave with respect to each master to which it is connected, and parameter assignments.
<*my_ip*>_generation.rpt	IP or Qsys generation log file. A summary of the messages during IP generation.
<*my_ip*>.debuginfo	Contains post-generation information. Used to pass System Console and Bus Analyzer Toolkit information about the Qsys interconnect. The Bus Analysis Toolkit uses this file to identify debug components in the Qsys interconnect.
<*my_ip*>.qip	Contains all the required information about the IP component to integrate and compile the IP component in the Quartus Prime software.
<*my_ip*>.csv	Contains information about the upgrade status of the IP component.
<*my_ip*>.bsf	A Block Symbol File (**.bsf**) representation of the IP variation for use in Quartus Prime Block Diagram Files (**.bdf**).
<*my_ip*>.spd	Required input file for `ip-make-simscript` to generate simulation scripts for supported simulators. The **.spd** file contains a list of files generated for simulation, along with information about memories that you can initialize.
<*my_ip*>.ppf	The Pin Planner File (**.ppf**) stores the port and node assignments for IP components created for use with the Pin Planner.
<*my_ip*>_bb.v	You can use the Verilog black-box (**_bb.v**) file as an empty module declaration for use as a black box.
<*my_ip*>_inst.v or **_inst.vhd**	HDL example instantiation template. You can copy and paste the contents of this file into your HDL file to instantiate the IP variation.

File Name	Description
<*my_ip*>.regmap	If the IP contains register information, the **.regmap** file generates. The **.regmap** file describes the register map information of master and slave interfaces. This file complements the **.sopcinfo** file by providing more detailed register information about the system. This enables register display views and user customizable statistics in System Console.
<*my_ip*>.svd	Allows HPS System Debug tools to view the register maps of peripherals connected to HPS within a Qsys system. During synthesis, the **.svd** files for slave interfaces visible to System Console masters are stored in the **.sof** file in the debug section. System Console reads this section, which Qsys can query for register map information. For system slaves, Qsys can access the registers by name.
<*my_ip*>.v or **<*my_ip*>.vhd**	HDL files that instantiate each submodule or child IP core for synthesis or simulation.
mentor/	Contains a ModelSim® script **msim_setup.tcl** to set up and run a simulation.
aldec/	Contains a Riviera-PRO script **rivierapro_setup.tcl** to setup and run a simulation.
/synopsys/vcs **/synopsys/vcsmx**	Contains a shell script **vcs_setup.sh** to set up and run a VCS® simulation. Contains a shell script **vcsmx_setup.sh** and **synopsys_ sim.setup** file to set up and run a VCS MX® simulation.
/cadence	Contains a shell script **ncsim_setup.sh** and other setup files to set up and run an NCSIM simulation.
/submodules	Contains HDL files for the IP core submodule.
***<IP submodule>*/**	For each generated IP submodule directory, Qsys generates **/synth** and **/sim** sub-directories.

Scripting IP Core Generation

You can use the `qsys-script` and `qsys-generate` utilities to define and generate an IP core variation outside of the Quartus Prime GUI.

To parameterize and generate an IP core at the command-line, follow these steps:

1. Run `qsys-script` to execute a Tcl script that instantiates the IP and sets desired parameters:

```
qsys-script --script=<script_file>.tcl
```

2. Run `qsys-generate` to generate the IP core variation:

```
qsys-generate <IP variation file>.qsys
```

Note: Creating an IP generation script is an advanced feature that requires access to special IP core parameters. For more information about creating an IP generation script, contact your Altera sales representative.

Table 4: qsys-generate Command-Line Options

Option	Usage	Description
`<1st arg file>`	Required	The name of the **.qsys** system file to generate.
`--synthesis=<VERILOG|VHDL>`	Optional	Creates synthesis HDL files that Qsys uses to compile the system in a Quartus Prime project. You must specify the preferred generation language for the top-level RTL file for the generated Qsys system.
`--block-symbol-file`	Optional	Creates a Block Symbol File (**.bsf**) for the Qsys system.
`--simulation=<VERILOG|VHDL>`	Optional	Creates a simulation model for the Qsys system. The simulation model contains generated HDL files for the simulator, and may include simulation-only features. You must specify the preferred simulation language.
`--testbench=<SIMPLE|STANDARD>`	Optional	Creates a testbench system that instantiates the original system, adding bus functional models (BFMs) to drive the top-level interfaces. When you generate the system, the BFMs interact with the system in the simulator.
`--testbench-simulation=<VERILOG|VHDL>`	Optional	After you create the testbench system, you can create a simulation model for the testbench system.

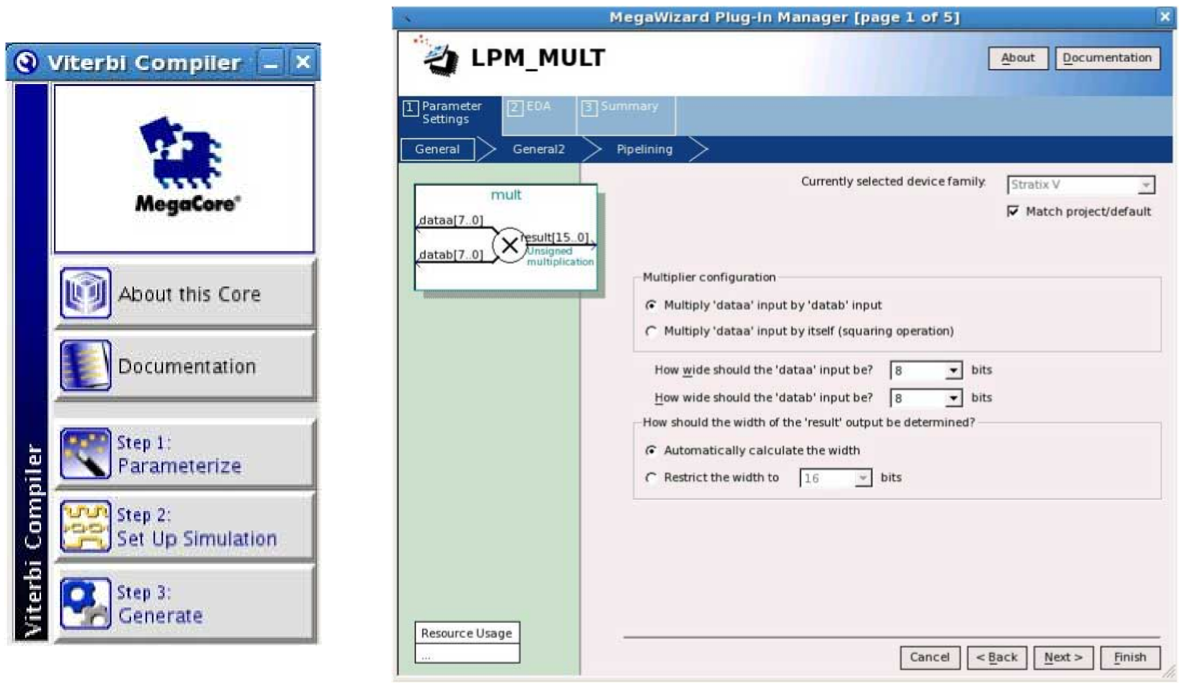
Option	Usage	Description
`--search-path=<value>`	Optional	If you omit this command, Qsys uses a standard default path. If you provide this command, Qsys searches a comma-separated list of paths. To include the standard path in your replacement, use "`$`", for example, "`/extra/dir,$`".
`--jvm-max-heap-size=<value>`	Optional	The maximum memory size that Qsys uses for allocations when running `qsys-generate`. You specify the value as `<size><unit>`, where `unit` is m (or M) for multiples of megabytes or g (or G) for multiples of gigabytes. The default value is 512m.
`--family=<value>`	Optional	Specifies the device family.
`--part=<value>`	Optional	Specifies the device part number. If set, this option overrides the `--family` option.
`--allow-mixed-language-simulation`	Optional	Enables a mixed language simulation model generation. If true, if a preferred simulation language is set, Qsys uses a `fileset` of the component for the simulation model generation. When false, which is the default, Qsys uses the language specified with `--file-set=<value>` for all components for simulation model generation. The current version of the ModelSim-Altera simulator supports mixed language simulation.

For command-line help listing all options for these executables, type <*executable name*> `--help`

Generating IP Cores (Legacy Editors)

Some IP cores use a legacy version of the parameter editor for configuration and generation. Use the following steps to configure and generate an IP variation using a legacy parameter editor.

Figure 7: Legacy Parameter Editors



Currently selected device family: Stratix V

[✓] Match project/default

Multiplier configuration

(●) Multiply 'dataa' input by 'datab' input

( ) Multiply 'dataa' input by itself (squaring operation)

How wide should the 'dataa' input be? 8 bits

How wide should the 'datab' input be? 8 bits

How should the width of the 'result' output be determined?

(●) Automatically calculate the width

( ) Restrict the width to 16 bits

Note: The legacy parameter editor generates a different output file structure than the latest parameter editor. Refer to *Specifying IP Core Parameters and Options* for configuration of IP cores that use the latest parameter editor.

1. In the IP Catalog (**Tools** > **IP Catalog**), locate and double-click the name of the IP core to customize. The parameter editor appears.
2. Specify a top-level name and output HDL file type for your IP variation. This name identifies the IP core variation files in your project. Click **OK**. Do not include spaces in IP variation names or paths.
3. Specify the parameters and options for your IP variation in the parameter editor. Refer to your IP core user guide for information about specific IP core parameters.
4. Click **Finish** or **Generate** (depending on the parameter editor version). The parameter editor generates the files for your IP variation according to your specifications. Click **Exit** if prompted when generation is complete. The parameter editor adds the top-level **.qip** file to the current project automatically.

   Note: For devices released prior to Arria 10 devices, the generated **.qip** and **.sip** files must be added to your project to represent IP and Qsys systems. To manually add an IP variation generated with legacy parameter editor to a project, click **Project** > **Add/Remove Files in Project** and add the IP variation **.qip** file.

Related Information

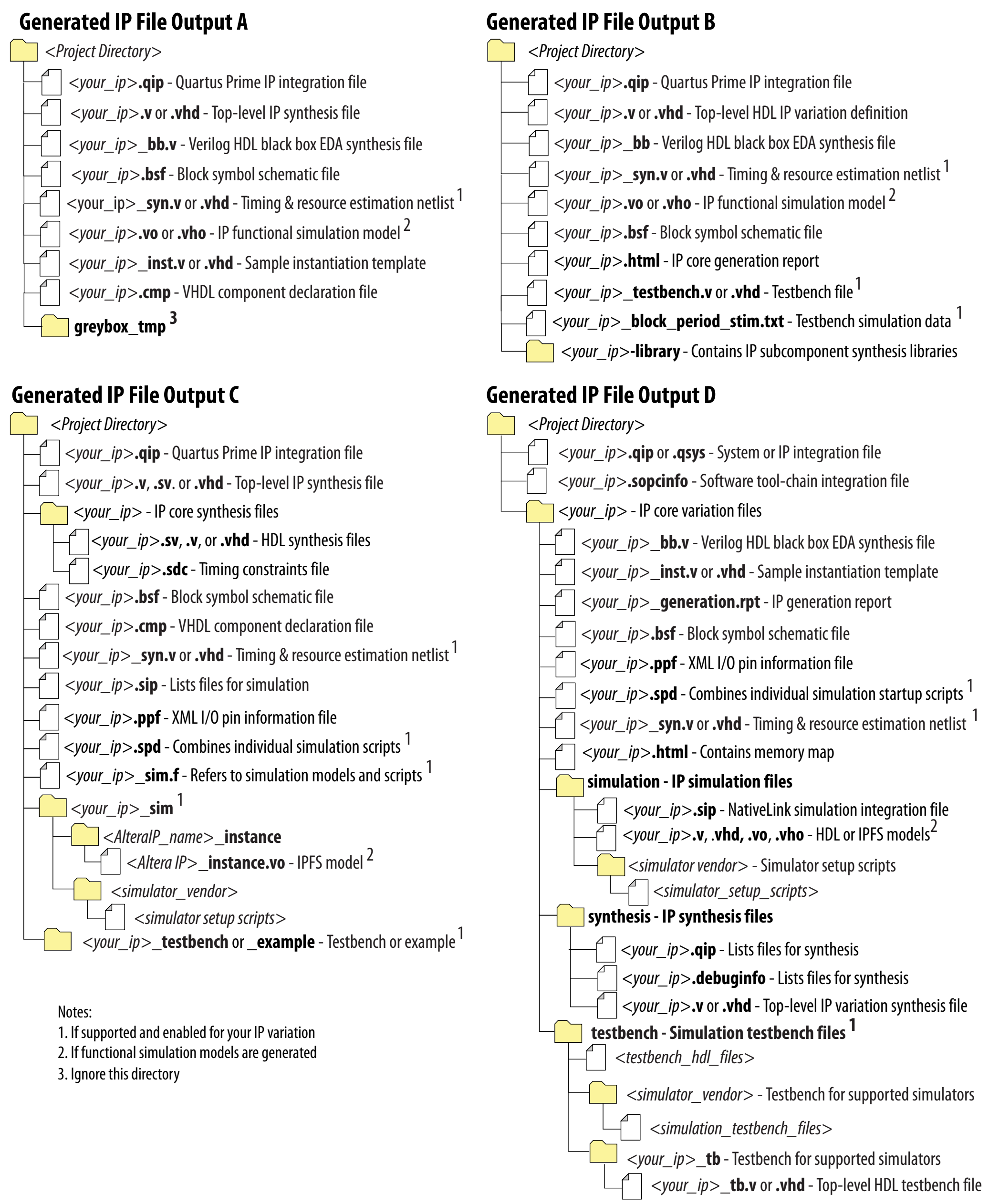
- IP User Guide Documentation

- Altera IP Release Notes

Files Generated for Altera IP Cores (Legacy Parameter Editors)

The Quartus Prime software generates one of the following output file structures for Altera IP cores that use a legacy parameter editor.

Figure 8: IP Core Generated Files (Legacy Parameter Editors)


Generated IP File Output A
<Project Directory>
<your_ip>.qip - Quartus Prime IP integration file
<your_ip>.v or .vhd - Top-level IP synthesis file
<your_ip>_bb.v - Verilog HDL black box EDA synthesis file
<your_ip>.bsf - Block symbol schematic file
<your_ip>_syn.v or .vhd - Timing & resource estimation netlist 1
<your_ip>.vo or .vho - IP functional simulation model 2
<your_ip>_inst.v or .vhd - Sample instantiation template
<your_ip>.cmp - VHDL component declaration file
greybox_tmp 3
Generated IP File Output B
<Project Directory>
<your_ip>.qip - Quartus Prime IP integration file
<your_ip>.v or .vhd - Top-level HDL IP variation definition
<your_ip>_bb - Verilog HDL black box EDA synthesis file
<your_ip>_syn.v or .vhd - Timing & resource estimation netlist 1
<your_ip>.vo or .vho - IP functional simulation model 2
<your_ip>.bsf - Block symbol schematic file
<your_ip>.html - IP core generation report
<your_ip>_testbench.v or .vhd - Testbench file 1
<your_ip>_block_period_stim.txt - Testbench simulation data 1
<your_ip>-library - Contains IP subcomponent synthesis libraries
Generated IP File Output C
<Project Directory>
<your_ip>.qip - Quartus Prime IP integration file
<your_ip>.v, .sv. or .vhd - Top-level IP synthesis file
<your_ip> - IP core synthesis files
<your_ip>.sv, .v, or .vhd - HDL synthesis files
<your_ip>.sdc - Timing constraints file
<your_ip>.bsf - Block symbol schematic file
<your_ip>.cmp - VHDL component declaration file
<your_ip>_syn.v or .vhd - Timing & resource estimation netlist 1
<your_ip>.sip - Lists files for simulation
<your_ip>.ppf - XML I/O pin information file
<your_ip>.spd - Combines individual simulation scripts 1
<your_ip>_sim.f - Refers to simulation models and scripts 1
<your_ip>_sim 1
<AlteraIP_name>_instance
<Altera IP>_instance.vo - IPFS model 2
<simulator_vendor>
<simulator setup scripts>
<your_ip>_testbench or _example - Testbench or example 1
Notes:
1. If supported and enabled for your IP variation
2. If functional simulation models are generated
3. Ignore this directory
Generated IP File Output D
<Project Directory>
<your_ip>.qip or .qsys - System or IP integration file
<your_ip>.sopcinfo - Software tool-chain integration file
<your_ip> - IP core variation files
<your_ip>_bb.v - Verilog HDL black box EDA synthesis file
<your_ip>_inst.v or .vhd - Sample instantiation template
<your_ip>_generation.rpt - IP generation report
<your_ip>.bsf - Block symbol schematic file
<your_ip>.ppf - XML I/O pin information file
<your_ip>.spd - Combines individual simulation startup scripts 1
<your_ip>_syn.v or .vhd - Timing & resource estimation netlist 1
<your_ip>.html - Contains memory map
simulation - IP simulation files
<your_ip>.sip - NativeLink simulation integration file
<your_ip>.v, .vhd, .vo, .vho - HDL or IPFS models 2
<simulator vendor> - Simulator setup scripts
<simulator_setup_scripts>
synthesis - IP synthesis files
<your_ip>.qip - Lists files for synthesis
<your_ip>.debuginfo - Lists files for synthesis
<your_ip>.v or .vhd - Top-level IP variation synthesis file
testbench - Simulation testbench files 1
<testbench_hdl_files>
<simulator_vendor> - Testbench for supported simulators
<simulation_testbench_files>
<your_ip>_tb - Testbench for supported simulators
<your_ip>_tb.v or .vhd - Top-level HDL testbench file

Note: For devices released prior to Arria 10 devices, the generated **.qip** and **.sip** files must be added to your project to represent IP and Qsys systems. To manually add an IP variation to a Quartus Prime project, click **Project** > **Add/Remove Files in Project** and add only the IP variation **.qip** or **.qsys** file, but not both, to the project. Do not manually add the top-level HDL file to the project.

Modifying an IP Variation

After generating an IP core variation, you can modify its parameters in the parameter editor. Use any of the following methods to modify an IP variation in the parameter editor.

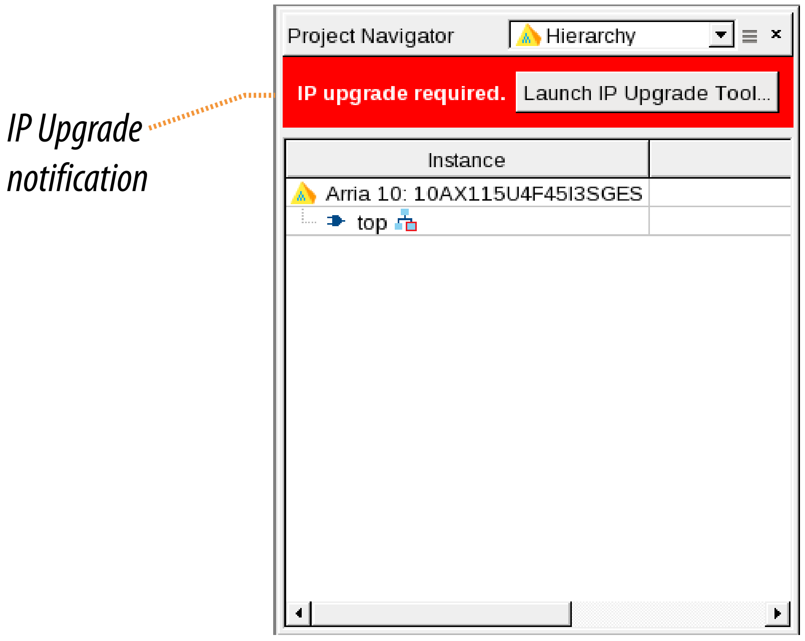
Table 5: Modifying an IP Variation

Menu Command	Action
File > **Open**	Select the top-level HDL (**.v**, or **.vhd**) IP variation file to launch the parameter editor and modify the IP variation. Regenerate the IP variation to implement your changes.
View > **Utility Windows** > **Project Navigator** > **IP Components**	Double-click the IP variation to launch the parameter editor and modify the IP variation. Regenerate the IP variation to implement your changes.
Project > **Upgrade IP Components**	Select the IP variation and click **Upgrade in Editor** to launch the parameter editor and modify the IP variation. Regenerate the IP variation to implement your changes.

Upgrading IP Cores

IP core variants generated with a previous version or different edition of the Quartus Prime software may require upgrading before use in the current version or edition of the Quartus Prime software. When you open a project containing outdated IP, the Project Navigator displays a banner indicating the IP upgrade status. Click **Launch IP Upgrade Tool**, or **Project** > **Upgrade IP Components** to upgrade outdated IP cores.

Figure 9: IP Upgrade Alert in Project Navigator



Icons in the **Upgrade IP Components** dialog box indicate when IP upgrade is required, optional, or unsupported for IP cores in your design. You must upgrade IP cores that require upgrade before you can compile the IP variation in the current version of the Quartus Prime software.

The upgrade process preserves the original IP variation file in the project directory as <*my_variant*>_**BAK.qsys**.

Note: Upgrading IP cores may append a unique identifier to the original IP core entity name(s), without similarly modifying the IP instance name. There is no requirement to update these entity references in any supporting Quartus Prime file; such as the Quartus Prime Settings File (**.qsf**), Synopsys Design Constraints File (**.sdc**), or SignalTap File (**.stp**), if these files contain instance names. The Quartus Prime software reads only the instance name and ignores the entity name in paths that specify both names. Use only instance names in assignments.

Table 6: IP Core Upgrade Status

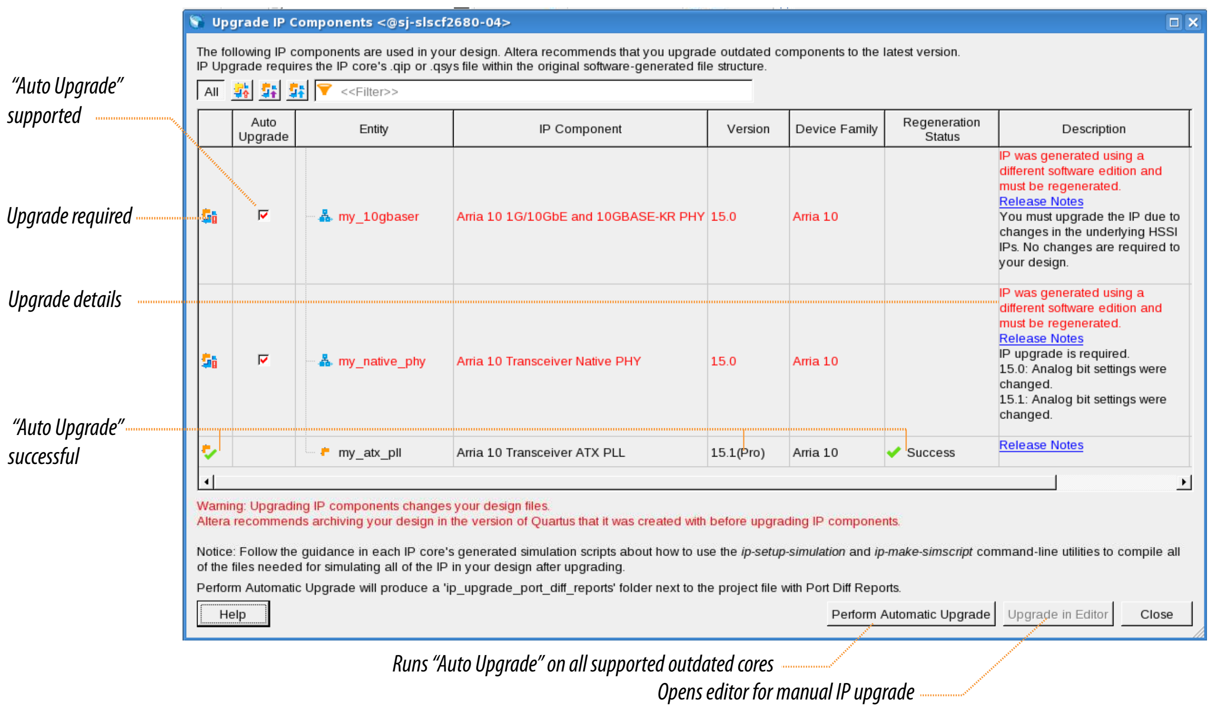
IP Core Status	Description
IP Upgraded	Your IP variation uses the latest version of the IP core.
IP Upgrade Optional	Upgrade is optional for this IP variation in the current version of the Quartus Prime software. You can upgrade this IP variation to take advantage of the latest development of this IP core. Alternatively you can retain previous IP core characteristics by declining to upgrade. Refer to the Description for details about IP core version differences. If you do not upgrade the IP, the IP variation synthesis and simulation files are unchanged and you cannot modify parameters until upgrading.
IP Upgrade Required	You must upgrade the IP variation before compiling in the current version of the Quartus Prime software. Refer to the Description for details about IP core version differences.
IP Upgrade Unsupported	Upgrade of the IP variation is not supported in the current version of the Quartus Prime software due to incompatibility with the current version of the Quartus Prime software. You are prompted to replace the unsupported IP core with a supported equivalent IP core from the IP Catalog. Refer to the Description for details about IP core version differences and links to Release Notes.
IP End of Life	Altera designates the IP core as end-of-life status. You may or may not be able to edit the IP core in the parameter editor. Support for this IP core discontinues in future releases of the Quartus Prime software.

IP Core Status	Description
IP Upgrade Mismatch Warning	Warning of non-critical IP core differences in migrating IP to another device family.

Follow these steps to upgrade IP cores:

1. In the latest version of the Quartus Prime software, open the Quartus Prime project containing an outdated IP core variation. The **Upgrade IP Components** dialog automatically displays the status of IP cores in your project, along with instructions for upgrading each core. Click **Project** > **Upgrade IP Components** to access this dialog box manually.
2. To upgrade one or more IP cores that support automatic upgrade, ensure that the **Auto Upgrade** option is turned on for the IP core(s), and then click **Perform Automatic Upgrade**. The **Status** and **Version** columns update when upgrade is complete. Example designs provided with any Altera IP core regenerate automatically whenever you upgrade an IP core.
3. To manually upgrade an individual IP core, select the IP core and then click **Upgrade in Editor** (or simply double-click the IP core name. The parameter editor opens, allowing you to adjust parameters and regenerate the latest version of the IP core.

Figure 10: Upgrading IP Cores


Upgrade IP Components <@sj-slscf2680-04>
All
<<Filter>>
"Auto Upgrade" supported
Upgrade required
Upgrade details
"Auto Upgrade" successful
Help
Perform Automatic Upgrade
Upgrade in Editor
Close
Runs "Auto Upgrade" on all supported outdated cores
Opens editor for manual IP upgrade

The following IP components are used in your design. Altera recommends that you upgrade outdated components to the latest version.
IP Upgrade requires the IP core's .qip or .qsys file within the original software-generated file structure.

	Auto Upgrade	Entity	IP Component	Version	Device Family	Regeneration Status	Description
	✓	my_10gbaser	Arria 10 1G/10GbE and 10GBASE-KR PHY	15.0	Arria 10		IP was generated using a different software edition and must be regenerated. Release Notes You must upgrade the IP due to changes in the underlying HSSI IPs. No changes are required to your design.
	✓	my_native_phy	Arria 10 Transceiver Native PHY	15.0	Arria 10		IP was generated using a different software edition and must be regenerated. Release Notes IP upgrade is required. 15.0: Analog bit settings were changed. 15.1: Analog bit settings were changed.
		my_atx_pll	Arria 10 Transceiver ATX PLL	15.1(Pro)	Arria 10	✓ Success	Release Notes

Warning: Upgrading IP components changes your design files.
Altera recommends archiving your design in the version of Quartus that it was created with before upgrading IP components.

Notice: Follow the guidance in each IP core's generated simulation scripts about how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design after upgrading.

Perform Automatic Upgrade will produce a 'ip_upgrade_port_diff_reports' folder next to the project file with Port Diff Reports.

Note: IP cores older than Quartus Prime software version 12.0 do not support upgrade. Altera verifies that the current version of the Quartus Prime software compiles the previous version of each IP core. The *Altera IP Release Notes* reports any verification exceptions for Altera IP cores. Altera does not verify compilation for IP cores older than the previous two releases.

Related Information

Altera IP Release Notes

Upgrading IP at Command-Line

You can upgrade an IP core at the command-line if the IP core supports auto upgrade. IP cores that do not support automatic upgrade do not support command-line upgrade.

- To upgrade a single IP core at the command-line, type the following command:

```
quartus_sh –ip_upgrade –variation_files <my_ip>.<qsys,.v, .vhd> <quartus_project>

Example:
quartus_sh -ip_upgrade -variation_files mega/pll25.qsys hps_testx
```

- To simultaneously upgrade multiple IP cores at the command-line, type the following command:

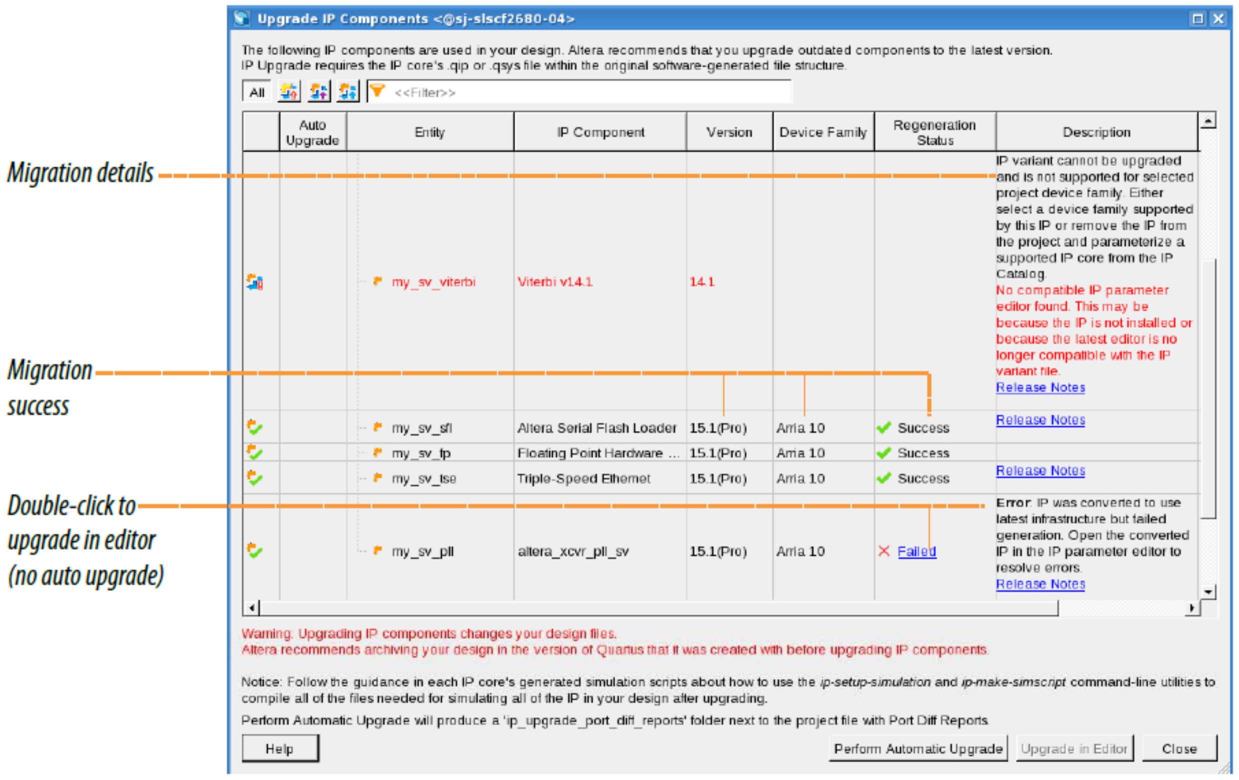
```
quartus_sh –ip_upgrade –variation_files “<my_ip1>.<qsys,.v, .vhd>>;
<my_ip_filepath/my_ip2>.<hdl>”  <quartus_project>

Example:
quartus_sh -ip_upgrade -variation_files "mega/pll_tx2.qsys;mega/pll3.qsys"
hps_testx
```

Migrating IP Cores to a Different Device

IP migration allows you to target the latest device families with IP originally generated for a different device. Most Altera IP cores support automatic migration. Some IP cores require manual IP regeneration for migration. Some IP cores do not support device migration and must be replaced in your design. The text and icons in the **Upgrade IP Components** dialog box identifies the migration support for each IP core in the design.

Note: Migration of some IP cores requires installed support for the original and migration device families.

Figure 11: IP Core Device Migration


Migration details
Migration success
Double-click to upgrade in editor (no auto upgrade)
Upgrade IP Components <@sj-slscf2680-04>
All
<<Filter>>
Help
Perform Automatic Upgrade
Upgrade in Editor
Close

The following IP components are used in your design. Altera recommends that you upgrade outdated components to the latest version.
IP Upgrade requires the IP core's .qip or .qsys file within the original software-generated file structure.

	Auto Upgrade	Entity	IP Component	Version	Device Family	Regeneration Status	Description
		my_sv_viterbi	Viterbi v14.1	14.1			IP variant cannot be upgraded and is not supported for selected project device family. Either select a device family supported by this IP or remove the IP from the project and parameterize a supported IP core from the IP Catalog. No compatible IP parameter editor found. This may be because the IP is not installed or because the latest editor is no longer compatible with the IP variant file. Release Notes
		my_sv_sfl	Altera Serial Flash Loader	15.1(Pro)	Arria 10	✓ Success	Release Notes
		my_sv_fp	Floating Point Hardware ...	15.1(Pro)	Arria 10	✓ Success	
		my_sv_tse	Triple-Speed Ethernet	15.1(Pro)	Arria 10	✓ Success	Release Notes
		my_sv_pll	altera_xcvr_pll_sv	15.1(Pro)	Arria 10	✗ Failed	Error: IP was converted to use latest infrastructure but failed generation. Open the converted IP in the IP parameter editor to resolve errors. Release Notes

Warning: Upgrading IP components changes your design files.
Altera recommends archiving your design in the version of Quartus that it was created with before upgrading IP components.

Notice: Follow the guidance in each IP core's generated simulation scripts about how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design after upgrading.

Perform Automatic Upgrade will produce a 'ip_upgrade_port_diff_reports' folder next to the project file with Port Diff Reports.

1. Click **File** > **Open Project** and open the Quartus Prime project containing IP for migration to another device in the original version of the Quartus Prime software. If prompted, click **Yes** to change to a supported device family.
2. To specify a different target device for migration, click **Assignments** > **Device** and select the target device family.
3. To display IP cores requiring migration, click **Project** > **Upgrade IP Components**. The **Description** field prompts you to run auto update or double-click IP cores for migration.
4. To migrate one or more IP cores that support automatic upgrade, ensure that the **Auto Upgrade** option is turned on for the IP core(s), and then click **Perform Automatic Upgrade**. The **Status** and **Version** columns update when upgrade is complete.
5. To migrate an IP core that does not support automatic upgrade, double-click the IP core name, and then click **OK**. The parameter editor appears. If the parameter editor specifies a **Currently selected device family**, turn off **Match project/default**, and then select the new target device family.
6. Click **Generate HDL**, and then confirm the **Synthesis** and **Simulation** file options. Verilog HDL is the default output file format specified. If your original IP core was generated for VHDL, select **VHDL** to retain the original output format.
7. Click **Finish** to complete migration of the IP core. Click **OK** if you are prompted to overwrite IP core files. The **Device Family** column displays the new target device name when migration is complete. The migration process replaces <*my_ip*>**.qip** with the <*my_ip*>**.qsys** top-level IP file in your project.

Note: If migration does not replace <*my_ip*>**.qip** with <*my_ip*>**.qsys**, click **Project > Add/Remove Files in Project** to replace the file in your project.

8. Review the latest parameters in the parameter editor or generated HDL for correctness. IP migration may change ports, parameters, or functionality of the IP core. During migration, the IP core's HDL generates into a library that is different from the original output location of the IP core. Update any assignments that reference outdated locations. If your upgraded IP core is represented by a symbol in a supporting Block Design File schematic, replace the symbol with the newly generated <*my_ip*>**.bsf** after migration.

    Note: The migration process may change the IP variation interface, parameters, and functionality. This may require you to change your design or to re-parameterize your variant after the **Upgrade IP Components** dialog box indicates that migration is complete. The **Description** field identifies IP cores that require design or parameter changes.

Related Information

Altera IP Release Notes

Troubleshooting IP or Qsys System Upgrade

The **Upgrade IP Components** dialog box reports the version and status of each IP core and Qsys system following upgrade or migration. If any upgrade or migration fails, the **Upgrade IP Components** dialog box provides information to help you resolve any errors.

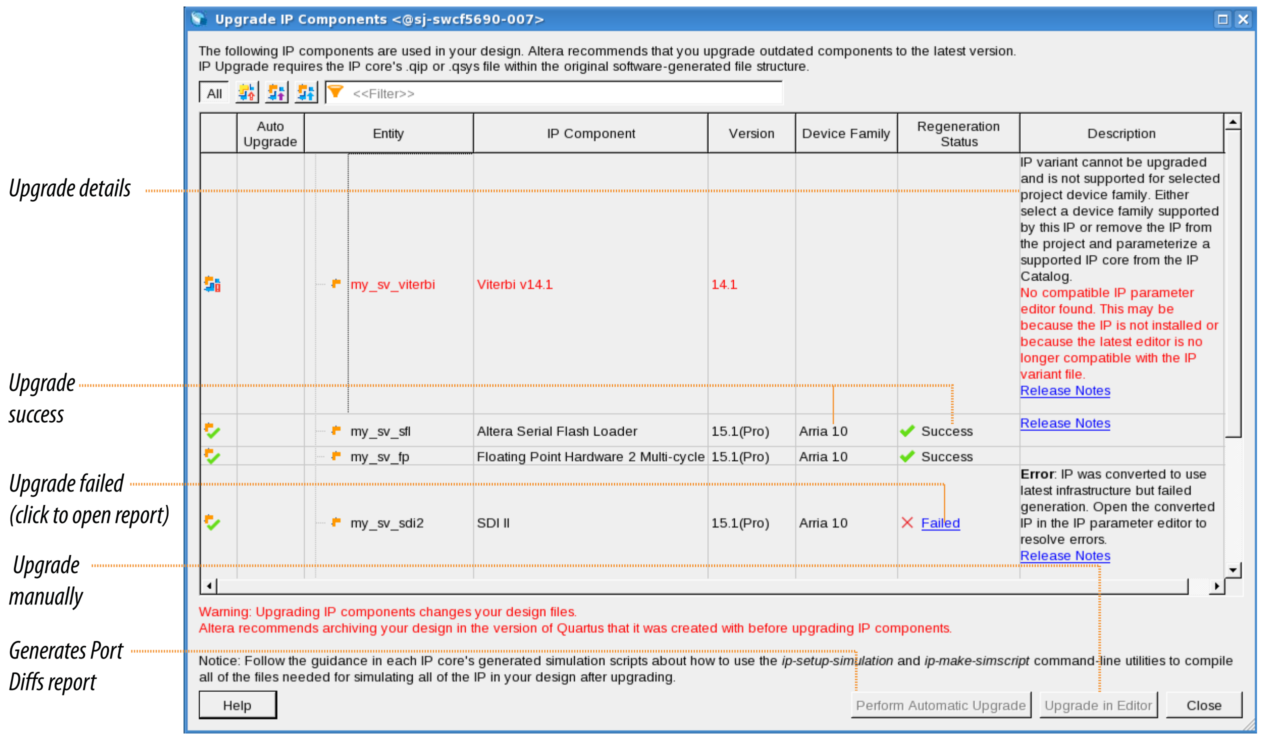
Note: Make sure that your IP variation names or paths do not include spaces. Spaces can be problematic for IP generation.

During automatic or manual upgrade, the Messages window dynamically displays upgrade information for each IP core or Qsys system. You can use the following information to help you resolve any upgrade errors following upgrade or migration.

Table 7: IP Upgrade Error Information

Upgrade IP Components Field	Description
Regeneration Status	Displays the "Success" or "Failed" status of each upgrade or migration. Click the status of any failed upgrade to open a detailed **IP Upgrade Report**.
Version	Dynamically updates to the new version number when upgrade is successful. The text is red when upgrade is required.
Device Family	Dynamically updates to the new device family when migration is successful. The text is red when upgrade is required.
Description	Summarizes IP release information and displays actionable, corrective action for resolving upgrade or migration failures. Follow these instructions to resolve upgrade failures. Click the **Release Notes** link for the latest known issues about the Altera IP core.
Perform Automatic Upgrade	Runs automatic upgrade on all IP cores that support auto upgrade. Also, automatically generates a <*Project Directory*>**/ip_upgrade_port_diff_report** report for IP cores or Qsys systems that fail upgrade. Review these reports to determine any port differences between the current and previous IP core version.

Figure 12: Resolving Upgrade Errors


Upgrade details
Upgrade success
Upgrade failed (click to open report)
Upgrade manually
Generates Port Diffs report
Upgrade IP Components <@sj-swcf5690-007>
All
<<Filter>>
Help
Perform Automatic Upgrade
Upgrade in Editor
Close

The following IP components are used in your design. Altera recommends that you upgrade outdated components to the latest version.
IP Upgrade requires the IP core's .qip or .qsys file within the original software-generated file structure.

	Auto Upgrade	Entity	IP Component	Version	Device Family	Regeneration Status	Description
		my_sv_viterbi	Viterbi v14.1	14.1			IP variant cannot be upgraded and is not supported for selected project device family. Either select a device family supported by this IP or remove the IP from the project and parameterize a supported IP core from the IP Catalog. No compatible IP parameter editor found. This may be because the IP is not installed or because the latest editor is no longer compatible with the IP variant file. Release Notes
✓		my_sv_sfl	Altera Serial Flash Loader	15.1(Pro)	Arria 10	✓ Success	Release Notes
✓		my_sv_fp	Floating Point Hardware 2 Multi-cycle	15.1(Pro)	Arria 10	✓ Success	
✓		my_sv_sdi2	SDI II	15.1(Pro)	Arria 10	✗ Failed	**Error**: IP was converted to use latest infrastructure but failed generation. Open the converted IP in the IP parameter editor to resolve errors. Release Notes

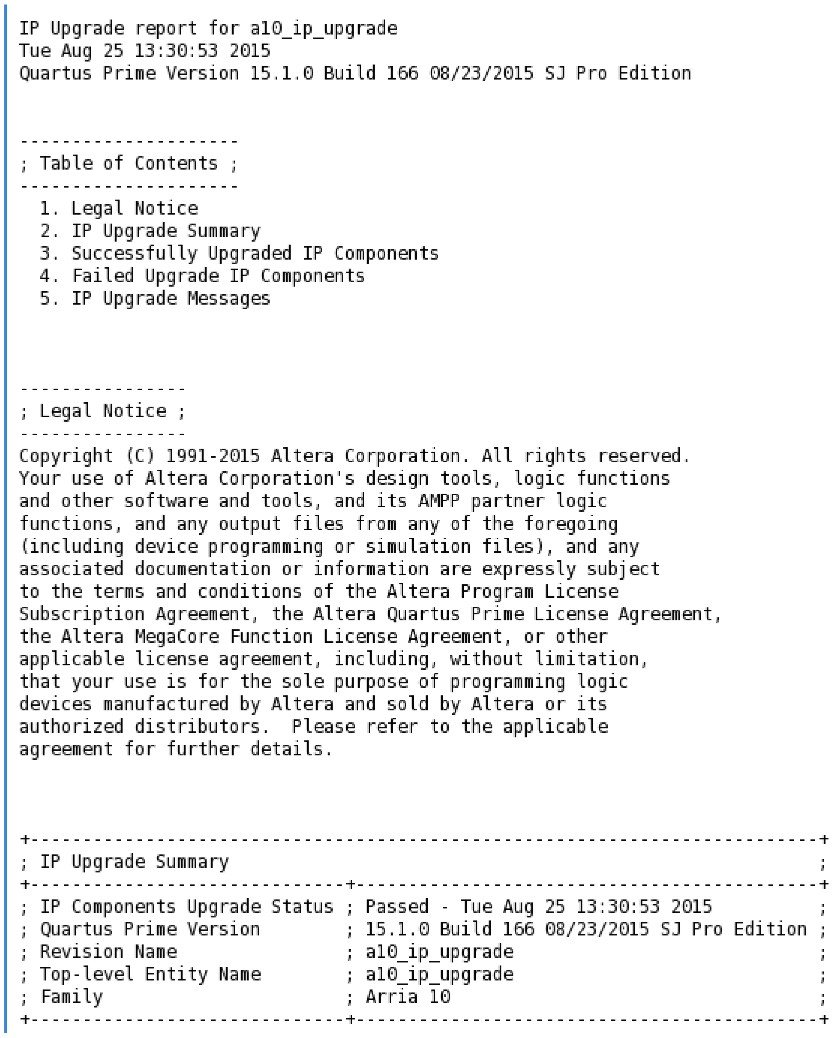
Warning: Upgrading IP components changes your design files.
Altera recommends archiving your design in the version of Quartus that it was created with before upgrading IP components.

Notice: Follow the guidance in each IP core's generated simulation scripts about how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design after upgrading.

Use the following techniques to resolve errors if your Altera IP core or Qsys system "Failed" to upgrade versions or migrate to another device. Review and implement the instructions in the **Description** field, including one or more of the following:

1. If the IP variant is not supported in the current version of the software, right-click the component and click **Remove IP Component from Project**. Replace this IP core or Qsys system with one supported in the current version of the software.
2. If the IP variant is not supported by the current target device, select a supported device family for the project, or replace the IP variant with a suitable replacement that supports your target device.
3. If an upgrade or migration fails, click **Failed** in the **Regeneration Status** field to display and review details of the **IP Upgrade Report**. Click the **Release Notes** link for the latest known issues about the Altera IP core. Use this information to determine the nature of the upgrade or migration failure and make corrections before upgrade.

Figure 13: IP Upgrade Report



```
IP Upgrade report for a10_ip_upgrade
Tue Aug 25 13:30:53 2015
Quartus Prime Version 15.1.0 Build 166 08/23/2015 SJ Pro Edition


---------------------
; Table of Contents ;
---------------------
  1. Legal Notice
  2. IP Upgrade Summary
  3. Successfully Upgraded IP Components
  4. Failed Upgrade IP Components
  5. IP Upgrade Messages



----------------
; Legal Notice ;
----------------
Copyright (C) 1991-2015 Altera Corporation. All rights reserved.
Your use of Altera Corporation's design tools, logic functions
and other software and tools, and its AMPP partner logic
functions, and any output files from any of the foregoing
(including device programming or simulation files), and any
associated documentation or information are expressly subject
to the terms and conditions of the Altera Program License
Subscription Agreement, the Altera Quartus Prime License Agreement,
the Altera MegaCore Function License Agreement, or other
applicable license agreement, including, without limitation,
that your use is for the sole purpose of programming logic
devices manufactured by Altera and sold by Altera or its
authorized distributors.  Please refer to the applicable
agreement for further details.



+-----------------------------------------------------------------------------+
; IP Upgrade Summary                                                          ;
+-------------------------------+---------------------------------------------+
; IP Components Upgrade Status ; Passed - Tue Aug 25 13:30:53 2015            ;
; Quartus Prime Version         ; 15.1.0 Build 166 08/23/2015 SJ Pro Edition ;
; Revision Name                 ; a10_ip_upgrade                              ;
; Top-level Entity Name         ; a10_ip_upgrade                              ;
; Family                        ; Arria 10                                    ;
+-------------------------------+---------------------------------------------+
```

4. Run **Perform Automatic Upgrade** to automatically generate an **IP Ports Diff** report for each IP core or Qsys system that fails upgrade. Review the reports to determine any port differences between the current and previous IP core version. Then, click **Upgrade in Editor** to make specific port changes and regenerate your IP core or Qsys system.
5. If your IP core or Qsys system does not support **Perform Automatic Upgrade**, click **Upgrade in Editor** to resolve errors and regenerate the component in the parameter editor.

Simulating Altera IP Cores

The Quartus Prime software supports RTL and gate-level simulation of Altera IP cores in supported EDA simulators. The Quartus Prime software generates simulation files for each IP core during IP generation, including the functional simulation model, any testbench (or example design), and vendor-specific simulator setup scripts for each IP core. You can use the functional simulation model and the testbench or example design generated with your IP core for simulation. The IP generation output also includes scripts to compile and run any testbench. The generated scripts list all models or libraries required to simulate your IP core.

The Quartus Prime software provides integration with your simulator and supports multiple simulation flows, including your own scripted and custom simulation flows. Whichever flow you chose, IP core simulation involves the following steps:

1. Generate simulation model, testbench (or example design), and simulator setup script files
2. Set up your simulator environment and any simulation script(s)
3. Compile simulation model libraries
4. Run your simulator

The Quartus Prime software integrates with your preferred simulation environment. This section describes how to setup and run typical scripted and NativeLink simulation flows.

Related Information

Simulating Altera Designs

Simulation Flows

The Quartus® Prime software supports various method for integrating your supported simulator into the design flow.

Table 8: Simulation Flows

Simulation Flow	Description
Scripted Simulation Flows	Scripted simulation supports custom control of all aspects of simulation, such as custom compilation commands, or multi-pass simulation flows. Altera recommends use of a version-independent top-level simulation script that "sources" Quartus Prime-generated IP simulation setup scripts. Use the `ip-setup-simulation` utility to combine and update individual simulator setup scripts, as described in this document.
NativeLink Simulation Flow	NativeLink automates Quartus Prime integration with your EDA simulator. You can setup NativeLink to generate simulation scripts, compile simulation libraries, and automatically launch your simulator following design compilation. You can specify your own compilation, elaboration, and simulation scripts for testbench and simulation model files that have not been analyzed by the Quartus Prime software. Do not use NativeLink if you require direct control over every aspect of simulation. **Note:** The Quartus Prime Pro Edition software does not support NativeLink RTL simulation.
Specialized Simulation Flows	Altera supports specialized simulation flows for specific design variations, including the following: • For simulation of Altera example designs, refer to the documentation for the example design or to the IP core user guide. • For simulation of Qsys designs, refer to *Creating a System with Qsys*. • For simulation of designs that include the Nios II embedded processor, refer to *Simulating a Nios II Embedded Processor*.

Related Information

- Scripting IP Simulation on page 25
- Using NativeLink Simulation
- Generating a Combined Simulator Setup Script on page 26
- IP User Guide Documentation
- Creating a System with Qsys
- Simulating a Nios II Embedded Processor

Generating IP Simulation Files

The Quartus Prime software optionally generates the functional simulation model, any testbench (or example design), and vendor-specific simulator setup scripts when you generate the IP core from the parameter editor or command-line including for each IP core. Use the following to control the generation of IP simulation files:

- Click **Assignment** > **Settings** to specify your supported simulator and options for IP simulation file generation.
- Click **Tools** > **IP Catalog** to parameterize a new IP variation, enable generation of simulation files, and generate the IP core synthesis and simulation files.
- Click **View** > **Utility Windows** > **Project Navigator** > **IP Components** to edit parameters and regenerate synthesis or simulation for an existing IP core variation.

Table 9: Altera IP Simulation Files

File Type	Description	File Name
Simulator setup scripts	Vendor-specific scripts to compile, elaborate, and simulate Altera IP models and simulation model library files. Source these files from your top-level simulation script, or edit these files to compile, elaborate, and simulate your design and testbench.	**<*my_dir*>/aldec/rivierapro_setup.tcl** **<*my_dir*>/cadence/ncsim_setup.sh** **<*my_dir*>/mentor/msim_setup.tcl** **<*my_dir*>/synopsys/vcs/vcs_setup.sh**
Simulation IP File	Contains IP core simulation library mapping information. To use NativeLink, you must add the **.qip** and **.sip** files generated for IP or Qsys systems to your project.	**<*design name*>.sip**
Qsys System File	Contains IP core simulation library mapping information. To use NativeLink for Arria 10 devices and later, you must add the **.qsys** file generated for IP or Qsys system to your project.	**<*design name*>.sip**
IP functional simulation models	IP functional simulation models are cycle-accurate VHDL or Verilog HDL models generated by the Quartus Prime software for some Altera IP cores. IP functional simulation models support fast functional simulation of IP using industry-standard VHDL and Verilog HDL simulators.	**<*my_ip*>.vho** **<*my_ip*>.vo**

File Type	Description	File Name
IEEE encrypted models	Arria V, Cyclone V, Stratix V, and newer simulation model libraries and IP simulation models are provided in Verilog HDL and IEEE encrypted Verilog HDL. VHDL simulation of these models is supported using your simulator's co-simulation capabilities. IEEE encrypted Verilog HDL models are significantly faster than IP functional simulation models.	***<my_ip>.v***

Note: Altera IP supports a variety of simulation models, including simulation-specific IP functional simulation models and encrypted RTL models, and plain text RTL models. These are all cycle-accurate models. The models support fast functional simulation of your IP core instance using industry-standard VHDL or Verilog HDL simulators. For some cores, only the plain text RTL model is generated, and you can simulate that model. Use the simulation models only for simulation and not for synthesis or any other purposes. Using these models for synthesis creates a nonfunctional design.

Related Information

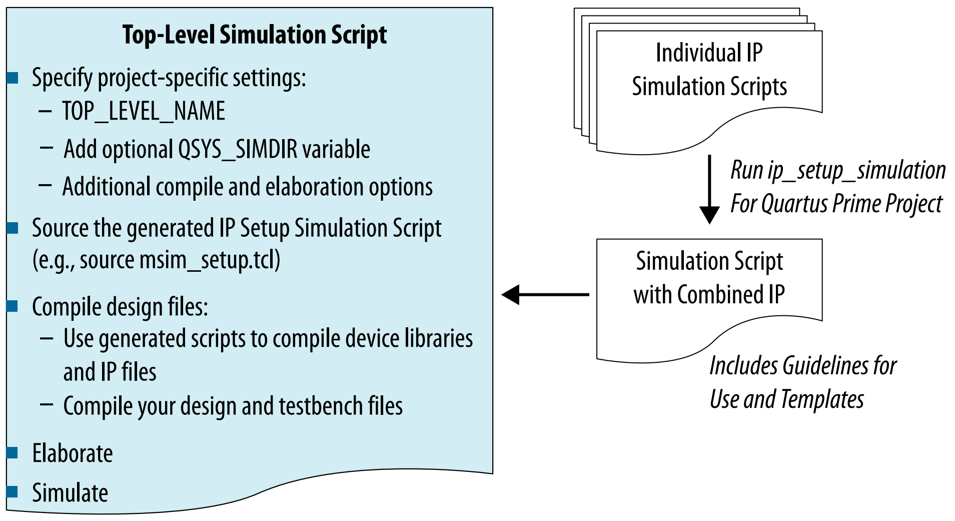
- **Generating IP Functional Simulation Models for 40nm Devices** on page 34
- **Generating IP Cores** on page 6
- **Modifying an IP Variation** on page 15
- **Upgrading IP Cores** on page 15

Scripting IP Simulation

The Quartus Prime software supports the use of scripts to automate simulation processing in your preferred simulation environment. You can use your preferred scripting methodology to control simulation.

Altera recommends the use of a version-independent top-level simulation script to control design, testbench, and IP core simulation. Because Quartus Prime-generated simulation file names may change after IP upgrade or regeneration, Altera recommends that your top-level simulation script "sources" any generated setup scripts, rather than using the generated setup scripts directly. You can use the `ip-setup-simulation` utility to generate or regenerate underlying setup scripts after any software or IP version upgrade or regeneration. Use of a top-level script and `ip-setup-simulation` eliminates the requirement to manually update simulation scripts.

Figure 14: Incorporating Generated Simulator Setup Scripts into a Top-Level Simulation Script


Top-Level Simulation Script
Specify project-specific settings:
– TOP_LEVEL_NAME
– Add optional QSYS_SIMDIR variable
– Additional compile and elaboration options
Source the generated IP Setup Simulation Script (e.g., source msim_setup.tcl)
Compile design files:
– Use generated scripts to compile device libraries and IP files
– Compile your design and testbench files
Elaborate
Simulate
Individual IP Simulation Scripts
Run ip_setup_simulation For Quartus Prime Project
Simulation Script with Combined IP
Includes Guidelines for Use and Templates

Generating a Combined Simulator Setup Script

The Quartus Prime software provides utilities to help you generate and update IP simulation scripts. You can use the `ip-setup-simulation` utility to generate a combined simulator setup script, for all Altera IP in your design, for each supported simulator. You can subsequently rerun `ip-setup-simulation` to automatically update the combined script. Each simulator's combined script file contains a rudimentary template that you can adapt for integration of the setup script into a top-level simulation script.

Table 10: Simulation Script Utilities

Utility	Syntax

`ip-setup-simulation`—Generates a combined, version-independent simulation script for all Altera IP cores in your project, and automates regeneration of the script after upgrading software or IP versions. Use the `compile-to-work` option to compile all simulation files into a single work library if your simulation environment requires that structure. Use the `--use-relative-paths` option to use relative paths whenever possible.	`ip-setup-simulation`
	`--quartus-project=<my proj>`
	`--output-directory=<my_dir>`
	`--use-relative-paths`
	`--compile-to-work`
	
	--use-relative-paths and --compile-to-work are optional. For command-line help listing all options for these executables, type: <*utility name*> --help.

`ip-make-simscript`—Generates a combined simulation script for all IP cores specified on the command line. Specify one or more **.spd** files and an output directory in the command. Running the script compiles IP simulation models into various simulation libraries.	`ip-make-simscript`
	`--spd=<ipA.spd,ipB.spd>`
	`--output-directory=<directory>`

Running ip-setup-simulation

To generate or update combined simulator setup scripts for all IP cores in your design, follow these steps:

1. Generate, regenerate, or upgrade one or more Altera IP core.
2. Run `ip-setup-simulation` on the project containing the IP core:

```
ip-setup-simulation --quartus-project=<my proj>.qpf
  --output-directory=<my dir>
  --use-relative-paths
```

3. To incorporate the simulator setup script into your top-level simulation script, refer to the template section in the generated simulator setup script as a guide to sourcing the generated script:
   a. Copy the specified template sections from the simulator-specific generated scripts and paste them into a new top-level file.
   b. Remove the comments at the beginning of each line from the copied template sections.
   c. Include the customizations required to match your design simulation requirements, for example:
      - Specify the `TOP_LEVEL_NAME` variable to the design's simulation top-level file. The top-level entity of your simulation is often a testbench that instantiates your design, and then your design instantiates IP cores and/or Qsys systems. Set the value of `TOP_LEVEL_NAME` to the top-level entity.
      - If necessary, set the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files.
      - Compile the top-level HDL file (e.g. a test program) and all other files in the design.
      - Specify any other changes, such as using the grep command-line utility to search a transcript file for error signatures, or e-mail a report.
4. To automatically update the combined IP simulation scripts, run `ip-setup-simulation` after any of the following events:
   - IP core initial generation or regeneration with new parameters
   - Upgrade of Quartus Prime software version
   - Upgrade of IP core version

Refer to the following topics for detailed steps for using the templates for each vendor.

Sourcing Aldec Simulator Setup Scripts

To incorporate generated Aldec simulation scripts into a top-level project simulation script, follow these steps:

1. The generated simulation script contains the following template lines. Cut and paste these lines into a new file. For example, **sim_top.tcl**.

```
# # Start of template
# # If the copied and modified template file is "aldec.do", run it as:
# # vsim -c -do aldec.do
# #
# # Source the generated sim script
# source rivierapro_setup.tcl
# # Compile eda/sim_lib contents first
# dev_com
# # Override the top-level name (so that elab is useful)
# set TOP_LEVEL_NAME top
# # Compile the standalone IP.
# com
# # Compile the user top-level
# vlog -sv2k5 ../../top.sv
# # Elaborate the design.
# elab
# # Run the simulation
# run
# # Report success to the shell
# exit -code 0
# # End of template
```

2. Delete the first two characters of each line (comment and space):

```
# Start of template
# If the copied and modified template file is "aldec.do", run it as:
# vsim -c -do aldec.do
#
# Source the generated sim script source rivierapro_setup.tcl
# Compile eda/sim_lib contents first dev_com
# Override the top-level name (so that elab is useful)
set TOP_LEVEL_NAME top
# Compile the standalone IP.
com
# Compile the user top-level vlog -sv2k5 ../../top.sv
# Elaborate the design.
elab
# Run the simulation
run
# Report success to the shell
exit -code 0
# End of template
```

3. Modify the `TOP_LEVEL_NAME` and compilation step appropriately, depending on the simulation's top-level file. For example:

```
set TOP_LEVEL_NAME sim_top
 vlog –sv2k5 ../../sim_top.sv
```

4. If necessary, add the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files. Specify any other changes required to match your design simulation requirements. The scripts offer variables to set compilation or simulation options. Refer to the generated script for details.
5. Run the new top-level script from the generated simulation directory:

```
vsim –c –do <path to sim_top>.tcl
```

Sourcing Cadence Simulator Setup Scripts

To incorporate generated Cadence IP simulation scripts into a top-level project simulation script, follow these steps:

1. The generated simulation script contains the following template lines. Cut and paste these lines into a new file. For example, **ncsim.sh**.

```
# # Start of template
# # If the copied and modified template file is "ncsim.sh", run it as:
# # ./ncsim.sh
# #
# # Do the file copy, dev_com and com steps
# source ncsim_setup.sh \
# SKIP_ELAB=1 \
# SKIP_SIM=1
#
# # Compile the top level module
# ncvlog -sv "$QSYS_SIMDIR/../top.sv"
#
# # Do the elaboration and sim steps
# # Override the top-level name
# # Override the user-defined sim options, so the simulation
# # runs forever (until $finish()).
# source ncsim_setup.sh \
# SKIP_FILE_COPY=1 \
# SKIP_DEV_COM=1 \
# SKIP_COM=1 \
# TOP_LEVEL_NAME=top \
# USER_DEFINED_SIM_OPTIONS=""
# # End of template
```

2. Delete the first two characters of each line (comment and space):

```
# Start of template
# If the copied and modified template file is "ncsim.sh", run it as:
# ./ncsim.sh
#
# Do the file copy, dev_com and com steps
source ncsim_setup.sh \
SKIP_ELAB=1 \
SKIP_SIM=1
# Compile the top level module
ncvlog -sv "$QSYS_SIMDIR/../top.sv"
# Do the elaboration and sim steps
# Override the top-level name
# Override the user-defined sim options, so the simulation
# runs forever (until $finish()).
source ncsim_setup.sh \
SKIP_FILE_COPY=1 \
SKIP_DEV_COM=1 \
SKIP_COM=1 \
TOP_LEVEL_NAME=top \
USER_DEFINED_SIM_OPTIONS=""
# End of template
```

3. Modify the `TOP_LEVEL_NAME` and compilation step appropriately, depending on the simulation's top-level file. For example:

```
TOP_LEVEL_NAME=sim_top \
 ncvlog -sv "$QSYS_SIMDIR/../top.sv"
```

4. If necessary, add the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files. Specify any other changes required to match your design simulation requirements. The scripts offer variables to set compilation or simulation options. Refer to the generated script for details.
5. Run the resulting top-level script from the generated simulation directory by specifying the path to **ncsim.sh**.

Sourcing ModelSim Simulator Setup Scripts

To incorporate generated ModelSim IP simulation scripts into a top-level project simulation script, follow these steps:

1. The generated simulation script contains the following template lines. Cut and paste these lines into a new file. For example, **sim_top.tcl**.

```
# # Start of template
# # If the copied and modified template file is "mentor.do", run it
# # as: vsim -c -do mentor.do
# #
# # Source the generated sim script
# source msim_setup.tcl
# # Compile eda/sim_lib contents first
# dev_com
# # Override the top-level name (so that elab is useful)
# set TOP_LEVEL_NAME top
# # Compile the standalone IP.
# com
# # Compile the user top-level
# vlog -sv ../../top.sv
# # Elaborate the design.
# elab
# # Run the simulation
# run -a
# # Report success to the shell
# exit -code 0
# # End of template
```

2. Delete the first two characters of each line (comment and space):

```
# Start of template
# If the copied and modified template file is "mentor.do", run it
# as: vsim -c -do mentor.do
#
# Source the generated sim script source msim_setup.tcl
# Compile eda/sim_lib contents first
dev_com
# Override the top-level name (so that elab is useful)
set TOP_LEVEL_NAME top
# Compile the standalone IP.
com
# Compile the user top-level vlog -sv ../../top.sv
# Elaborate the design.
elab
# Run the simulation
run -a
# Report success to the shell
exit -code 0
# End of template
```

3. Modify the `TOP_LEVEL_NAME` and compilation step appropriately, depending on the simulation's top-level file. For example:

```
set TOP_LEVEL_NAME sim_top vlog -sv ../../sim_top.sv
```

4. If necessary, add the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files. Specify any other changes required to match your design simulation requirements. The scripts offer variables to set compilation or simulation options. Refer to the generated script for details.
5. Run the resulting top-level script from the generated simulation directory:

```
vsim –c –do <path to sim_top>.tcl
```

Sourcing VCS Simulator Setup Scripts

To incorporate generated Synopsys VCS simulation scripts into a top-level project simulation script, follow these steps:

1. The generated simulation script contains these template lines. Cut and paste the lines preceding the "helper file" into a new executable file. For example, **synopsys_vcs.f**.

```
# # Start of template
# # If the copied and modified template file is "vcs_sim.sh", run it
# # as: ./vcs_sim.sh
# #
# # Override the top-level name
# # specify a command file containing elaboration options
# # (system verilog extension, and compile the top-level).
# # Override the user-defined sim options, so the simulation
# # runs forever (until $finish()).
# source vcs_setup.sh \
# TOP_LEVEL_NAME=top \
# USER_DEFINED_ELAB_OPTIONS="'-f ../../../synopsys_vcs.f'" \
# USER_DEFINED_SIM_OPTIONS=""
#
# # helper file: synopsys_vcs.f
# +systemverilogext+.sv
# ../../../top.sv
# # End of template
```

2. Delete the first two characters of each line (comment and space) for the **vcs.sh** file, as shown below:

```
# Start of template
# If the copied and modified template file is "vcs_sim.sh", run it
# as: ./vcs_sim.sh
#
# Override the top-level name
# specify a command file containing elaboration options
# (system verilog extension, and compile the top-level).
# Override the user-defined sim options, so the simulation
# runs forever (until $finish()).
source vcs_setup.sh \
TOP_LEVEL_NAME=top \
USER_DEFINED_ELAB_OPTIONS="'-f ../../../synopsys_vcs.f'" \
USER_DEFINED_SIM_OPTIONS=""
```

3. Delete the first two characters of each line (comment and space) for the **synopsys_vcs.f** file, as shown below:

```
# helper file: synopsys_vcs.f
 +systemverilogext+.sv
 ../../../top.sv
# End of template
```

4. Modify the `TOP_LEVEL_NAME` and compilation step appropriately, depending on the simulation's top-level file. For example:

```
TOP_LEVEL_NAME=sim_top \
```

5. If necessary, add the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files. Specify any other changes required to match your design simulation requirements. The scripts offer variables to set compilation or simulation options. Refer to the generated script for details.
6. Run the resulting top-level script from the generated simulation directory by specifying the path to **vcs_sim.sh**.

Sourcing VCS MX Simulator Setup Scripts

To incorporate generated Synopsys VCS MX simulation scripts for use in top-level project simulation scripts, follow these steps:

1. The generated simulation script contains these template lines. Cut and paste the lines preceding the "helper file" into a new executable file. For example, **vcsmx.sh**.

```
# # Start of template
# # If the copied and modified template file is "vcsmx_sim.sh", run
# # it as: ./vcsmx_sim.sh
# #
# # Do the file copy, dev_com and com steps
# source vcsmx_setup.sh \
# SKIP_ELAB=1 \
# SKIP_SIM=1
#
# # Compile the top level module vlogan +v2k
     +systemverilogext+.sv "$QSYS_SIMDIR/../top.sv"

# # Do the elaboration and sim steps
# # Override the top-level name
# # Override the user-defined sim options, so the simulation runs
# # forever (until $finish()).
# source vcsmx_setup.sh \
# SKIP_FILE_COPY=1 \
# SKIP_DEV_COM=1 \
# SKIP_COM=1 \
# TOP_LEVEL_NAME="'-top top'" \
# USER_DEFINED_SIM_OPTIONS=""
# # End of template
```

2. Delete the first two characters of each line (comment and space), as shown below:

```
# Start of template
# If the copied and modified template file is "vcsmx_sim.sh", run
# it as: ./vcsmx_sim.sh
#
# Do the file copy, dev_com and com steps
source vcsmx_setup.sh \
SKIP_ELAB=1 \
SKIP_SIM=1

# Compile the top level module
vlogan +v2k +systemverilogext+.sv "$QSYS_SIMDIR/../top.sv"

# Do the elaboration and sim steps
# Override the top-level name
# Override the user-defined sim options, so the simulation runs
# forever (until $finish()).
source vcsmx_setup.sh \
SKIP_FILE_COPY=1 \
SKIP_DEV_COM=1 \
SKIP_COM=1 \
TOP_LEVEL_NAME="'-top top'" \
USER_DEFINED_SIM_OPTIONS=""
# End of template
```

3. Modify the `TOP_LEVEL_NAME` and compilation step appropriately, depending on the simulation's top-level file. For example:

```
TOP_LEVEL_NAME=”-top sim_top’” \
```

4. Make the appropriate changes to the compilation of the your top-level file, for example:

```
vlogan +v2k +systemverilogext+.sv "$QSYS_SIMDIR/../sim_top.sv"
```

5. If necessary, add the `QSYS_SIMDIR` variable to point to the location of the generated IP simulation files. Specify any other changes required to match your design simulation requirements. The scripts offer variables to set compilation or simulation options. Refer to the generated script for details.
6. Run the resulting top-level script from the generated simulation directory by specifying the path to **vcsmx_sim.sh**.

Using NativeLink Simulation

The NativeLink feature integrates your EDA simulator with the Quartus Prime software and automates the following simulation steps:

- Set and reuse simulation settings
- Generate simulator-specific files and simulation scripts
- Compile Altera simulation libraries
- Launch your simulator automatically following Quartus Prime Analysis & Elaboration, Analysis & Synthesis, or after a full compilation.

Note: The Quartus Prime Pro Edition software does not support NativeLink RTL simulation. To use NativeLink for Arria 10 devices, you must add the **.qsys** file generated for the IP or Qsys system. To use NativeLink for all other device families, you must add to your project the **.qip** and **.sip** files generated for IP or Qsys systems.

Setting Up NativeLink Simulation

Before running simulation using the NativeLink flow, you must specify settings for your simulator in the Quartus Prime software. To specify simulation settings in the Quartus Prime software, follow these steps:

1. Open a Quartus Prime project.
2. Click **Tools** > **Options** and specify the location of your simulator executable file .

Table 11: Execution Paths for EDA Simulators

Simulator	Path
Mentor Graphics ModelSim-Altera	<drive letter>**:**\<simulator install path>\ **win32aloem** (Windows) /<simulator install path>**/bin** (Linux)
Mentor Graphics ModelSim Mentor Graphics QuestaSim	<drive letter>**:**\<simulator install path>**\win32** (Windows) <simulator install path>**/bin** (Linux)
Synopsys VCS/VCS MX	<simulator install path>**/bin** (Linux)
Cadence Incisive Enterprise	<simulator install path>**/tools/bin** (Linux)
Aldec Active-HDL Aldec Riviera-PRO	<drive letter>**:**\<simulator install path>**\bin** (Windows) <simulator install path>**/bin** (Linux)

3. Click **Assignments** > **Settings** and specify options on the **Simulation** page and **More NativeLink Settings** dialog box. Specify default options for simulation library compilation, netlist and tool command script generation, and for launching RTL or gate-level simulation automatically following Quartus Prime processing.
4. If your design includes a testbench, turn on **Compile test bench** and then click **Test Benches** to specify options for each testbench. Alternatively, turn on **Use script to compile testbench** and specify the script file.
5. If you want to use a script to setup simulation, turn on **Use script to setup simulation**.

Generating IP Functional Simulation Models for 40nm Devices

Altera provides IP functional simulation models for some Altera IP cores supporting 40nm Altera devices. To generate IP functional simulation models, follow these steps:

- Turn on the **Generate Simulation Model** option when parameterizing the IP core.
- When you simulate your design, compile only the **.vo** or **.vho** for these IP cores in your simulator. In this case you should not compile the corresponding HDL file. The encrypted HDL file supports synthesis by only the Quartus Prime software.

Note: Altera IP cores that do not require IP functional simulation models for simulation, do not provide the **Generate Simulation Model** option in the IP core parameter editor.

Note: Many recently released Altera IP cores support RTL simulation using IEEE Verilog HDL encryption. IEEE encrypted models are significantly faster than IP functional simulation models. You can simulate the models in both Verilog HDL and VHDL designs.

Related Information

AN 343: OpenCore Evaluation of AMPP Megafunctions

Synthesizing Altera IP Cores in Other EDA Tools

You can use supported EDA tools to synthesize a design that includes Altera IP cores. When you generate the IP core synthesis files for use with third-party EDA synthesis tools, you can optionally create an area and timing estimation netlist. To enable generation, turn on **Create timing and resource estimates for third-party EDA synthesis tools** when customizing your IP variation.

The area and timing estimation netlist describes the IP core connectivity and architecture, but does not include details about the true functionality. This information enables certain third-party synthesis tools to better report area and timing estimates. In addition, synthesis tools can use the timing information to achieve timing-driven optimizations and improve the quality of results.

The Quartus Prime software generates the **<*variant name*>_syn.v** netlist file in Verilog HDL format regardless of the output file format you specify. If you use this netlist for synthesis, you must include the IP core wrapper file **<*variant name*>.v** or **<*variant name*>.vhd** in your Quartus Prime project.

Related Information

Quartus Prime Integrated Synthesis

Instantiating IP Cores in HDL

You can instantiate an IP core directly in your HDL code by calling the IP core name and declaring its parameters, in the same manner as any other module, component, or subdesign. When instantiating an IP core in VHDL, you must include the associated libraries.

Accessing HDL Code Templates

The software includes code examples or templates for inferred RAMs, ROMs, shift registers, arithmetic functions, and DSP functions optimized for Altera devices. To access HDL code templates to define these IP cores in HDL, follow these steps:

1. Open a file in the text editor.
2. Click **Edit** > **Insert template**.
3. In the **Insert Template** dialog box, click the + icon to expand either the **Verilog HDL** category or the **VHDL** category, depending on the HDL you prefer.
4. Under **Full Designs**, expand the navigation tree to display the type of functions you want to infer.
5. Select the function to display the code in the Preview pane, and then click **Insert**.

Example Top-Level Verilog HDL Module

Verilog HDL ALTFP_MULT in Top-Level Module with One Input Connected to Multiplexer.

```
module MF_top (a, b, sel, datab, clock, result);
       input [31:0] a, b, datab;
       input clock, sel;
       output [31:0] result;
       wire [31:0] wire_dataa;

       assign wire_dataa = (sel)? a : b;
       altfp_mult inst1
(.dataa(wire_dataa), .datab(datab), .clock(clock), .result(result));

       defparam
               inst1.pipeline = 11,
               inst1.width_exp = 8,
               inst1.width_man = 23,
               inst1.exception_handling = "no";
endmodule
```

Example Top-Level VHDL Module

VHDL ALTFP_MULT in Top-Level Module with One Input Connected to Multiplexer.

```
library ieee;
use ieee.std_logic_1164.all;
library altera_mf;
use altera_mf.altera_mf_components.all;

entity MF_top is
        port (clock, sel  : in  std_logic;
              a, b, datab : in  std_logic_vector(31 downto 0);
              result      : out std_logic_vector(31 downto 0));
end entity;

architecture arch_MF_top of MF_top is
signal wire_dataa : std_logic_vector(31 downto 0);
begin

wire_dataa <= a when (sel = '1') else b;

inst1 : altfp_mult
        generic map    (
                   pipeline => 11,
                   width_exp => 8,
                   width_man => 23,
                   exception_handling => "no")
        port map (
                   dataa => wire_dataa,
                   datab => datab,
                   clock => clock,
                   result => result);
end arch_MF_top;
```

Document Revision History

This document has the following revision history.

Date	Version	Changes
2016.02.05	15.1.1	• Corrected list of files ip-make-simscript generates. • Removed incorrect statement about running ip-make-simscript. • Revised Incorporating IP Simulation Scripts in Top-Level Scripts graphic.
2015.11.02	15.1.0	• Added Generating Version-Agnostic IP Simulation Scripts topic. • Added example IP simulation script templates for supported simulators. • Added Incorporating IP Simulation Scripts in Top-Level Scripts topic. • Added Troubleshooting IP Upgrade topic. • Updated IP Catalog and parameter editor descriptions for GUI changes. • Updated IP upgrade and migration steps for latest GUI changes. • Updated Generating IP Cores process for GUI changes. • Updated Files Generated for IP Cores and Qsys system description. • Updated product name throughout.
2015.05.04	15.0.0	• The latest version of the ModelSim-Altera software supports native, mixed language (VHDL/Verilog HDL/SystemVerilog) co-simulation of plain text HDL. • Added qsys_script IP core instantiation information. • Described changes to generating and processing of instance and entity names. • Added description of upgrading IP cores at the command line. • Updated procedures for upgrading and migrating IP cores. • Gate level timing simulation supported only for Cyclone IV and Stratix IV devices.
2014.12.1	14.1.0	Added information about new **Assignments** > **Settings** > **IP Settings** that control frequency of synthesis file regeneration and automatic addition of IP files to the project.

Date	Version	Changes
2014.08.18	14.0a10.0	• Added information about specifying parameters for IP cores targeting Arria 10 devices. • Added information about the latest IP output for Quartus II version 14.0a10 targeting Arria 10 devices. • Added information about individual migration of IP cores to the latest devices. • Added information about editing existing IP variations.
June 2014	14.0.0	• Changed title from *Introduction to Megafunctions* to *Introduction to Altera IP Cores*. • Increased scope of document to include updated information about licensing, customizing, upgrading, and simulating all Altera IP cores. • Replaced MegaWizard Plug-In Manager with IP Catalog information.
May 2013	13.0 .1	• Reorganization of content into topics. • First tracking of changes in Document Revision History.