Titanium Interfaces User Guide

UG-TiINTF-v6.1
June 2025
www.efinixinc.com

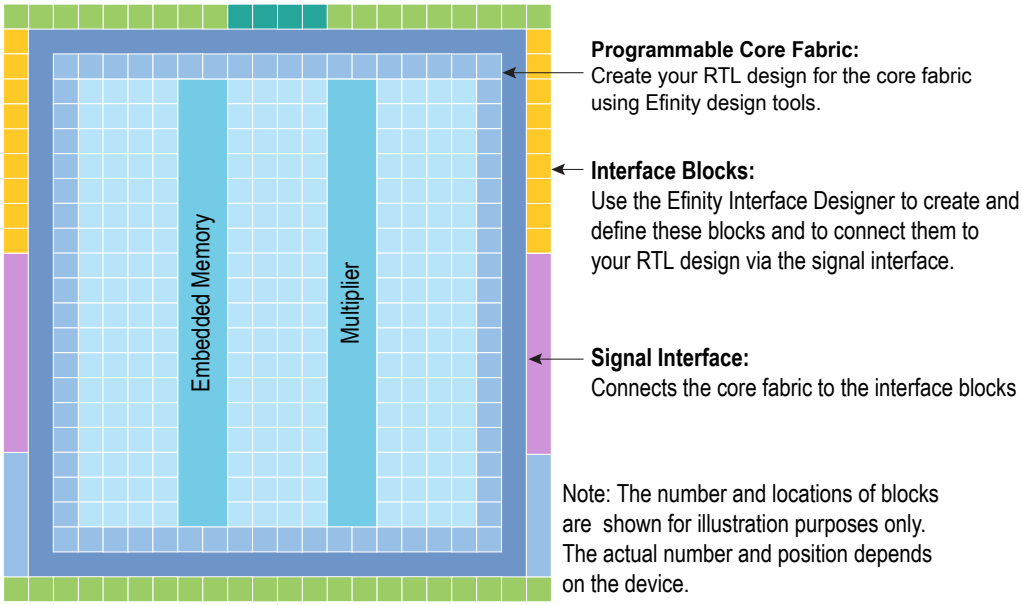
Contents

About the Interface Designer

Titanium FPGAs wrap a Quantum®-accelerated core with a periphery that sends signals out to the device pins. The core contains the logic, embedded memory, and multipliers. The device periphery includes blocks such as GPIO pins, LVDS, MIPI, DDR, and PLLs.

The tools in the Efinity® main window help you design the logic portion of your design. You use the Efinity Interface Designer to build the peripheral portion of your design.

Figure 1: Conceptual View of Interface Blocks



Chapter 1

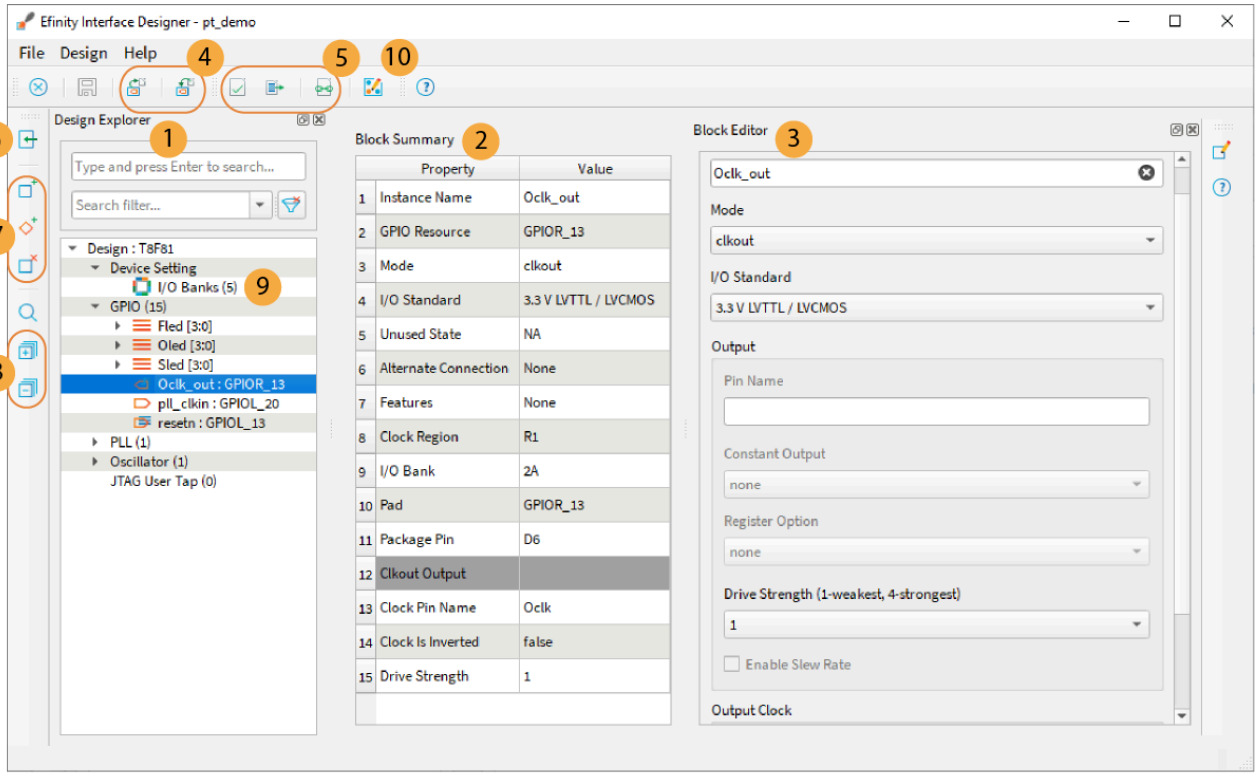
Get Oriented

Contents:

The Interface Designer has four main sections:

- *Design Explorer*—Provides a list view of the interface blocks you have in your design organized by block type. It also includes device-wide settings for the I/O banks and configuration options. Select a block to display it's summary and editor.
- *Block Summary*—Displays the current settings for the selected block.
- *Block Editor*—Provides options and settings for the selected block. The editor may have more than one tab, depending on the block.
- *Resource Assigner*—Provides an easy, tabular method for assigning resources. View by instance (default) or resource.

Figure 2: Interface Designer

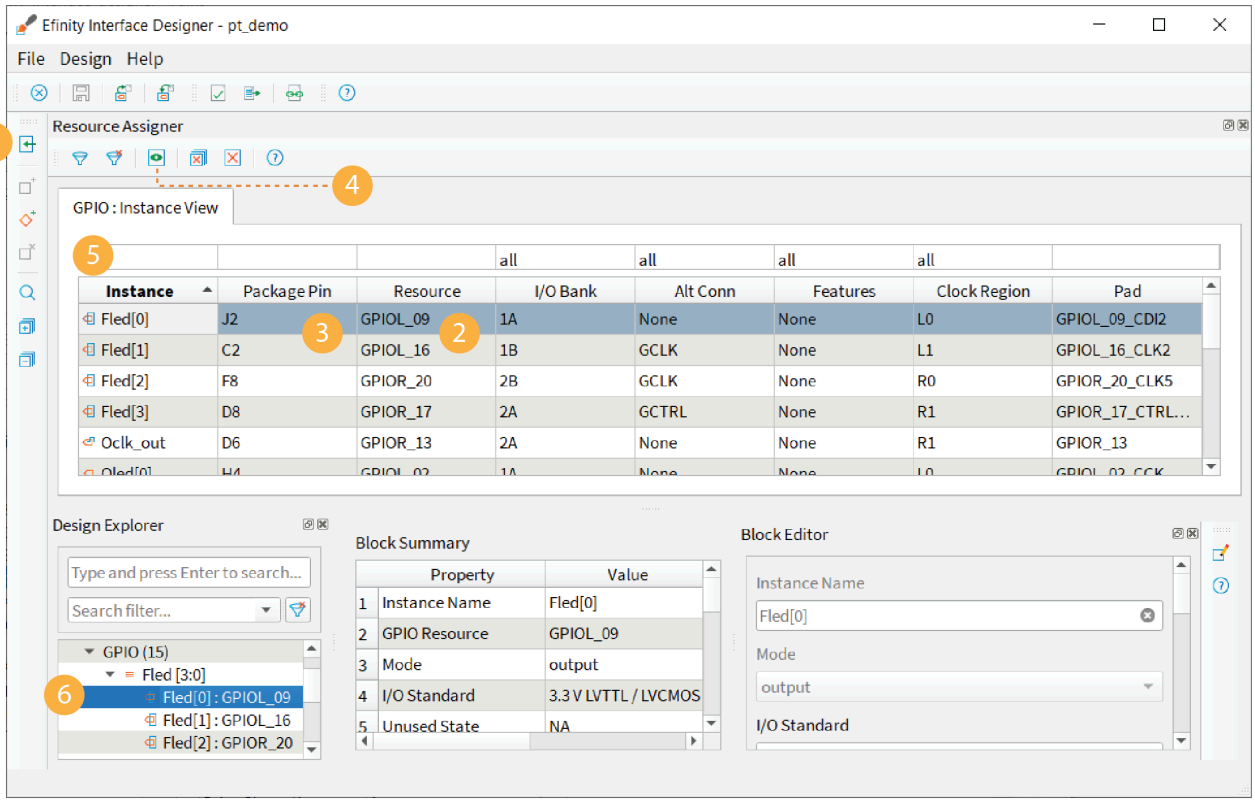


Notes:

1. The Design Explorer shows the interface blocks in your design. They are organized by block type.
2. The block summary shows the settings for the block selected in the Design Explorer.
3. Use the Block Editor to add or change settings for the interface block.
4. You can import or export GPIO resource assignments using a **.csv** or **.isf** file.
5. Use the project management tools to perform design checks, view reports, generate constraints, etc.
6. Click Show/Hide Resource Assigner to toggle a tabular view of assignments.
7. Use the block tools to add or delete blocks and buses.
8. Expand or collapse the Design Explorer folders.
9. The number in parentheses shows the number of used blocks.
10. The Package Planner lets you see the pins and assignments graphically.

When you first open the Interface Designer for your project, the Design Explorer shows the Device Settings folder (with default settings) and empty folders for the interface blocks your chosen device supports. You need to add blocks as required for your design.

Figure 3: Resource Assigner



Notes:

1. Show or hide the Resource Assigner.
2. Double-click in the Resource cell to open the list of available resources.
3. Double-click in the Package Pin cell to open the list of available pins.
4. Click the Switch View button to toggle between Instance View and Resource View.
5. Type in the filter cell above the column you want to filter.
6. Selecting a block in the Design Explorer highlights it in the Resource Assigner.

Interface Blocks

Titanium FPGAs support a variety of interface blocks. The available blocks differ depending on which FPGA you target and the package. You need to assign a resource for every block you use.

The following table describes the interface blocks supported in the Efinity® software.

Note: New package support is often added in patches. Refer to the Efinity Release Notes in the Support Center for the latest patch support.

Table 1: Titanium Interface Block Support by Package

Interface	Ti35	Ti60	Ti85	Ti90	Ti120	Ti135	Ti165	Ti180	Ti240	Ti375
DDR	–	–	All	J361, J484, G529	J361, J484, G529	All	All	J361 J484, M484, G529	All	All
GPIO	All	All	All	All	All	All	All	All	All	All
GPIO bus	All	All	All	All	All	All	All	All	All	All
HyperRAM	F100S3F2	F100S3F2	–	–	–	–	–	–	–	–
I/O bank	All	All	All	All	All	All	All	All	All	All
JTAG User TAP	All	All	All	All	All	All	All	All	All	All
LVDS TX LVDS RX Bidirectional LVDS	All	All	All	All	All	All	All	All	All	All
MIPI DPHY	–	–	All	J361, J484, L484	J361, J484, L484	All	N484, N900, N1156	J361, J484, L484, M484, J484D1	N484, N900, N1156	N484, N900, N1156
MIPI TX Lane MIPI RX Lane	All	All	All	All	All	All	All	All	All	All
PCI Express® Ethernet XGMII PMA Direct Ethernet SGMII	–	–	All	–	–	All	N484, N900, N1156	–	N484, N900. N1156	N484, N900, N1156
PLL (V3)	All	All	–	All	All	–	–	All	–	–
PLL (Fractional)	–	–	All	–	–	All	All	–	All	All

Interface	Ti35	Ti60	Ti85	Ti90	Ti120	Ti135	Ti165	Ti180	Ti240	Ti375
PLL SSC	–	–	All	J361, J484, L484	J361, J484, L484	All	N484, N900, N1156	J361, J484, L484, M484	N484, N900, N1156	N484, N900, N1156
Oscillator	All	All	All	All	All	All	All	All	All	All
Quad-Core RISC-V	–	–	All	–	–	All	All	–	All	All
SPI Flash	F100S3F2	F100S3F2	–	–	–	–	–	–	–	–

All interface blocks have an instance name that must be a unique identifier. When you add a new block, the Interface Designer gives the block a unique default name, which you can change.

Note: After you re-name the block, press Enter or click Save to save the name.

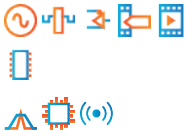
Pin names are the top-level ports of the design implemented in the core that connect to the interface block. These names must be legal Verilog HDL or VHDL identifiers.

Package/Interface Support Matrix

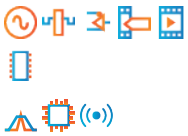
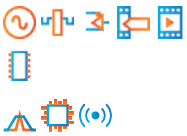
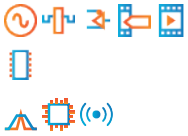
Some interfaces are only available in certain packages. The following table describes which interfaces are supported in specific FPGA/package combinations for the Efinity® software. Refer to the data sheet for package-dependent resources.

Note: New package support is often added in patches. Refer to the Efinity Release Notes in the Support Center for the latest patch support.

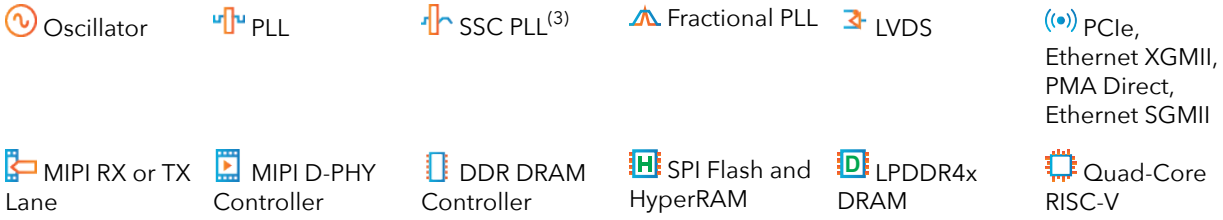
Table 2: Supported Titanium Interface/Package Combinations

Package	Ti35	Ti60	Ti90, Ti120, Ti180	Ti85, Ti135	Ti165, Ti240, Ti375
W64					
F100					
F100S3F2					
F225 F256					
G400					
J361, J484, M484[1]					
N441					

[1] Available for Ti180 only

Package	Ti35	Ti60	Ti90, Ti120, Ti180	Ti85, Ti135	Ti165, Ti240, Ti375
J484D1[2]					
L484					
N484					
C529					
G529					
N676					
N900, N1156					

Titanium Family Legend:



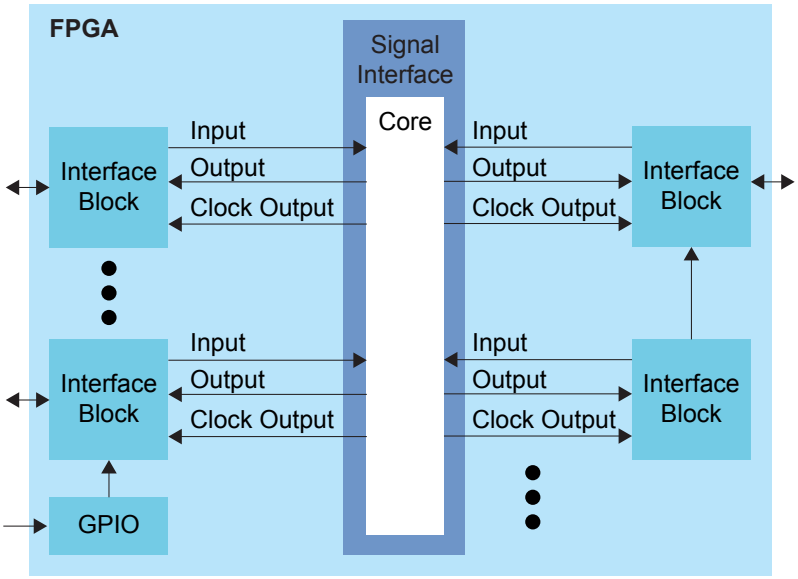
(2) Available for Ti180 only.
(3) The SSC PLL block and MIPI D-PHY TX Controller block cannot be used at the same time. Refer to **About the Spread-Spectrum Clocking PLL Interface** on page 167 for more information.

Interface Block Connectivity

The FPGA core fabric connects to the interface blocks through a signal interface. The interface blocks then connect to the package pins. The core connects to the interface blocks using three types of signals:

- *Input*—Input data or clock to the FPGA core
- *Output*—Output from the FPGA core
- *Clock output*—Clock signal from the core clock tree

Figure 4: Interface Block and Core Connectivity



GPIO blocks are a special case because they can operate in several modes. For example, in alternate mode the GPIO signal can bypass the signal interface and directly feed another interface block. So a GPIO configured as an alternate input can be used as a PLL reference clock without going through the signal interface to the core.

When designing for Titanium FPGAs, you create an RTL design for the core and also configure the interface blocks. From the perspective of the core, outputs from the core are inputs to the interface block and inputs to the core are outputs from the interface block.

The Efinity netlist always shows signals from the perspective of the core, so some signals do not appear in the netlist:

- GPIO used as reference clocks are not present in the RTL design, they are only visible in the interface block configuration of the Efinity® Interface Designer.
- The FPGA clock tree is connected to the interface blocks directly. Therefore, clock outputs from the core to the interface are not present in the RTL design, they are only part of the interface configuration (this includes GPIO configured as output clocks).

The following sections describe the different types of interface blocks. Signals and block diagrams are shown from the perspective of the interface, not the core.

Clocking Interface Blocks

The Interface Designer defines clock connections to the interfaces in the **<project name>.interface.csv** file, which allows the software to connects the clocks between the core and periphery. To use a clock in the periphery, simply use the clock name in the relevent field in the interface block.

- Clocks used in interfaces do not have to be used in the RTL.
- Clocks feeding the interfaces should not drive outputs in the RTL.

Note: This section assumes you understand how to create and use the GPIO and PLL blocks.

(See GPIO Interface on page 60 and PLL Interface on page 146).

The following sections show some common clocking examples to illustrate the general concept. There are other Interface Designer blocks that use clocks, and other modes for the GPIO and PLL blocks. You use the same method for referencing clocks in other blocks and modes.

GPIO Clocking I/O Register

To use a GPIO to clock an I/O register:

1. Instantiate a GPIO in input mode with connection type **gclk** or **rclk**. Put the name in **Input Clock tab > Pin Name**, e.g., `gpio_inst12`. This name is the clock name.
2. Instantiate a GPIO in output mode with **Output tab > Register Option > register**.
3. In **Output Clock tab > Pin Name**, enter the clock name from step 1 (`gpio_inst12`).

If this clock is not used in your RTL design, you do not have to include it. It's simply a clock from one interface block to another.

PLL Output Clocking I/O Register

To use a PLL output clock to clock an I/O register:

1. Instantiate a PLL and define an output clock, `pll_out1`. This name is the clock name.
2. Instantiate a GPIO in output mode with **Output tab > Register Option > register**.
3. In **Output Clock tab > Pin Name**, enter the clock name from step 1 (`pll_out1`).

If this clock is not used in your RTL design, you do not have to include it. It is simply a clock from one interface block to another.

Internal Clock Clocking I/O Register

To use an internally generated clock to clock an I/O register:

1. You create an internally generated clock, `div_clock`. This name is the clock name.
2. Instantiate a GPIO in output mode with **Output tab > Register Option > register**.
3. In **Output Clock tab > Pin Name**, enter the clock name from step 1 (`div_clock`).

You do not need to connect the internally generate clock to an output in the RTL. The Interface Designer connects to the clock network automatically.

PLL Output Driving a Pin

To use a PLL output clock to clock a pin:

1. Instantiate a PLL and define an output clock, `pll_out1`. This name is the clock name.
2. Instantiate a GPIO in **clkout** mode.
3. In **Output Clock tab > Pin Name**, enter the clock name from step 1 (`pll_out1`).

If this clock is not used in your RTL design, you do not have to include it. It is simply a clock from one interface block to another.

You can use this output pin to connect to other components on your board.

Designing an Interface

 Save

 Check Design

 Generate Report

 Generate Efinity Constraint Files

 Exit

Designing your interface is straightforward: add interface blocks, configure them, and then generate reports and constraints. The Efinity software uses the constraints during compilation to connect signals from the core to your interface.

Note: Refer to Create or Delete a Block on page 14 and Interface Blocks on page 9 for instructions on adding blocks and configuring them.

During the design process, you can generate reports, which are available in the Efinity® Results tab. When you generate reports, the software also saves your design.

Use the design checker to check the interface for errors and to ensure that your settings are valid. The Interface designer displays design issues in the message viewer window. You can also export design issues (**Design > Export Design Issues**) to generate a comma separated values (**.csv**) report to view the issues in a spreadsheet application. When you run the design checker, the software automatically saves your interface.

When you are done configuring your interface, click the Export Efinity Constraints Files button to export the interface constraints to your project. The software saves the design, checks it for errors, generates the interface reports and the interface constraint files.

Click Exit to close the Interface Designer and return to the Efinity® main window.

Note: You can leave the Interface Designer open while running the Efinity® software. However, if you make changes to the Efinity project, the Interface Designer is not updated until the next time you launch it.

Create or Delete a Block

 Create Block

 Create GPIO Bus

 Delete Block

To create a block:

1. Select the folder for the block type you want to create.
2. Click the Create Block button.

To create a GPIO bus, click the GPIO folder and then click the Create GPIO Bus button.

To delete a block, select the block name and click the Delete Block button.

Tip: Right-clicking a folder name opens a context-sensitive menu. From there you can choose **Create Block** (and **Create Bus** for GPIO).

Using the Resource Assigner

 Resource Assigner

 Switch View

 Clear Selected Resource

 Clear All Resources

 Show/Hide Filter

 Reset Filter

The Resource Assigner provides a tabular view of all GPIO resources in your chosen FPGA and information about them, such as whether they are used, the I/O bank, pad, and package pin, and the instance assigned to the resource.

- The **GPIO: Instance View** shows all GPIO instances in your project.
- The **GPIO: Resource View** shows all GPIO, LVDS, and MIPI RX or TX lane resources and the resources to which you assigned them.

Note: In the Efinity® software v2021.1, you can only view the resources used for LVDS and MIPI lanes in the Resource Assigner. You cannot change or assign resources in this view.

To assign a resource:

1. Open the Resource Assigner by clicking the Show/Hide Resource Assigner button. The software opens to the Instance View, which lists all instances in the design.

   Note: Click Switch View to toggle between instance view and resource view.

2. In instance view, you can assign pins or resources to the instance. Double-click in the table cell for the item you want to assign. The software displays a drop-down list of available selections.
3. Select an unused resource, instance, or pin.

   Note: If you select a used resource, instance, or pin, the software makes the new assignment, which replaces the previous assignment.

4. Press Enter.

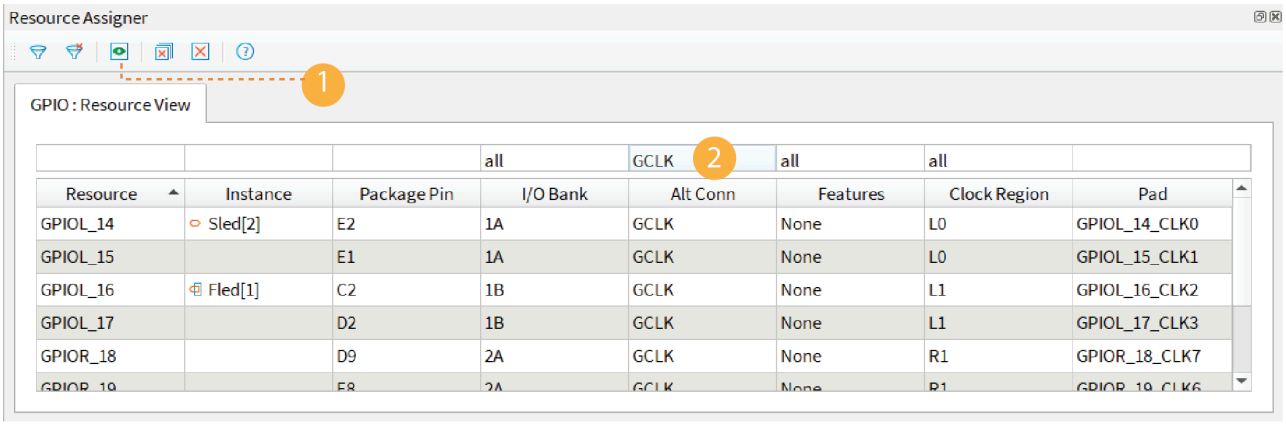
Note: Titanium: When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO pins in the same bank. This separation reduces noise. The Efinity software issues an error if you do not leave this separation.

Resource View

When assigning GPIO, sometimes you want to know which resource can be used as a global clock, global control, or other special function. You can look it up in the pin table for the FPGA and package you are targeting, but an easier way is to use the Resource View in the Resource Assigner.

1. Click the Switch View button to open the Resource View.
2. Double-click in the filter box above the **Alt Conn** column and choose the connection type, for example, **GCLK**.

Figure 5: Resource View



Importing and Exporting Assignments

Although it is nice to use a GUI for adding blocks, in some cases it may be easier to use another format. The Interface Designer lets you import and export assignments using an Interface Scripting File (**.isf**) or comma separated values (**.csv**) file.

When the software reads an imported **.isf**, it processes the entire imported file and shows any issues it found. The import only fails for catastrophic errors. The software:

- Creates new instances defined in the file that do not already exist in the GUI
- Overwrites assignments for existing instances with settings from the file
- Does not delete instances that are in the GUI but were not defined in the file

When the software reads an imported **.csv** file, it compares the imported assignments to the original assignments and reports any issues. If the software finds warnings, it displays them but allows you to finish the import. If it finds errors, it will not finish the import. When importing, the software:

- Deletes instances that you removed
- Creates newly defined instances
- Replaces instances you renamed with the new name

With the Efinity software v2025.1 and higher you can add an **.isf** to your project in the **Project Editor > Design tab**.

Learn more: For help understanding messages, refer to the "Design Check" topics in the interfaces user guides. These topics describe the messages the Interface Designer generates and gives suggestions on how to fix errors and warnings.

Interface Scripting File

The Interface Scripting File (**.isf**) contains all of the Python API commands to re-create your interface. You can export your design to an **.isf**, manipulate the file, and then re-import it back into the Efinity® software. Additionally, you can write your own **.isf** if desired.

In addition to using the API, you can export and import an **.isf** in the Interface Designer GUI. Click the Import GPIO or Export GPIO buttons and choose **Interface Scripting File (.isf)** under **Format**.

Example: Example Interface Scripting File

```
# Efinity Interface Configuration
# Version: 2020.M.138
# Date: 2020-06-26 14:22
#
# Copyright (C) 2017 - 2020 Efinix Inc. All rights reserved.
#
# Device: T8F81
# Package: 81-ball FBGA (final)
# Project: pt_demo
# Configuration mode: active (x1)
# Timing Model: C2 (final)

# Create instance
design.create_output_gpio("Fled",3,0)
design.create_inout_gpio("Sled",3,0)
design.create_output_gpio("Oled",3,0)
design.create_clockout_gpio("Oclk_out")
design.create_pll_input_clock_gpio("pll_clkin")
design.create_global_control_gpio("resetn")

# Set property, non-defaults
design.set_property("Fled","OUT_REG","REG")
design.set_property("Fled","OUT_CLK_PIN","Fclk")
design.set_property("Sled[0]","IN_PIN","")
design.set_property("Sled[0]","OUT_PIN","Sled[0]")
design.set_property("Sled[1]","IN_PIN","")
design.set_property("Sled[1]","OUT_PIN","Sled[1]")
design.set_property("Sled[2]","IN_PIN","")
design.set_property("Sled[2]","OUT_PIN","Sled[2]")
design.set_property("Sled[3]","IN_PIN","")
design.set_property("Sled[3]","OUT_PIN","Sled[3]")
design.set_property("Oclk_out","OUT_CLK_PIN","Oclk")

# Set resource assignment
design.assign_pkg_pin("Fled[0]","J2")
design.assign_pkg_pin("Fled[1]","C2")
design.assign_pkg_pin("Fled[2]","F8")
design.assign_pkg_pin("Fled[3]","D8")
design.assign_pkg_pin("Sled[0]","E6")
design.assign_pkg_pin("Sled[1]","G4")
design.assign_pkg_pin("Sled[2]","E2")
design.assign_pkg_pin("Sled[3]","G9")
design.assign_pkg_pin("Oled[0]","H4")
design.assign_pkg_pin("Oled[1]","J4")
design.assign_pkg_pin("Oled[2]","A5")
design.assign_pkg_pin("Oled[3]","C5")
design.assign_pkg_pin("Oclk_out","D6")
design.assign_pkg_pin("pll_clkin","C3")
design.assign_pkg_pin("resetn","F1")
```

.csv File for GPIO Blocks

For larger designs with lots of GPIO, it can be simpler to use a spreadsheet application to make assignments. The Resource Assigner allows you to import and export GPIO block assignments using a comma separated values (**.csv**) file. The **.csv** file includes the package pin and pad name, the instance name, and the mode. You can use this method for any type of GPIO, including LVDS pins used as GPIO or HSIO pins used as GPIO.

Table 3: Example GPIO .csv File

Package Pin-Pad Name	Instance Name	Mode
G5-GPIOL_00		
J4-GPIOL_01_SS_N		
H4-GPIOL02_CCK		
G4-GPIOL_03_CDI4	led[0]	output
F4-GPIOL04_CDI0	led[1]	output
J3-GPIOL_05_CDI5	rstn	input
H3-GPIOL_06_CDI1		
...		
(4)	led[6]	inout

When working with the **.csv** file:

- Add your assignments to the **Instance Name** and **Mode** columns.
- Do not modify the package pin-pad names.
- For the mode, specify: input, output, inout, clkout, or none

Note: You cannot make advanced settings such as alternate connections or registering. To make these settings, use the Block Editor.

When the software reads an imported **.csv** file, it performs a comparison between the **.csv** assignments and the original GPIO block assignments and reports any issues. If the software finds warnings, it displays them but allows you to finish the import. If it finds errors, it will not finish the import. When importing, the software:

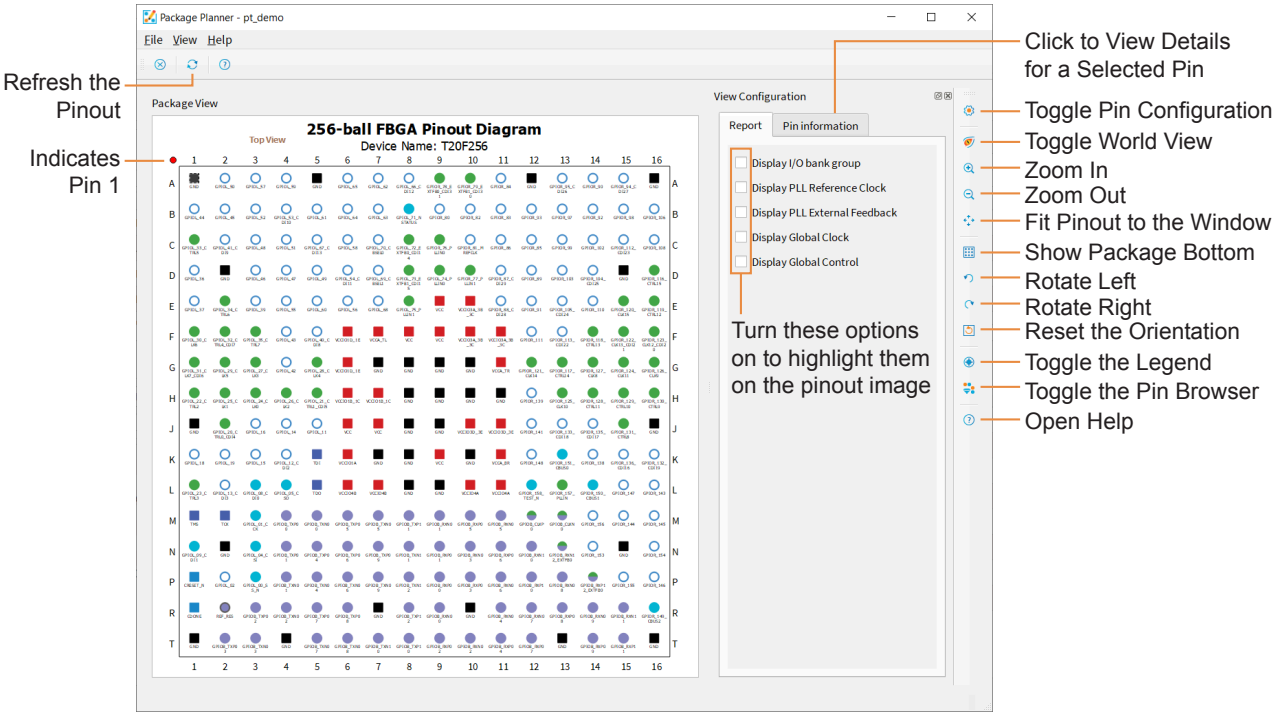
- Deletes instances that you removed
- Creates newly defined instances
- Replaces instances you renamed with the new name

(4) Unassigned instances have a blank field for the Package Pin-Pad Name column.

Viewing the Package Pinout

The Package Planner provides a visual representation of the FPGA package pins. Each pin is color coded by function (such as GPIO, configuration, power, etc.) letting you easily see which package pin has which function. Additionally, you can highlight I/O banks, PLL reference clocks, global clocks, and global controls so you can quickly find a specific pin that has the feature you need. This tool is helpful when planning how to map the signals in your design to package pins.

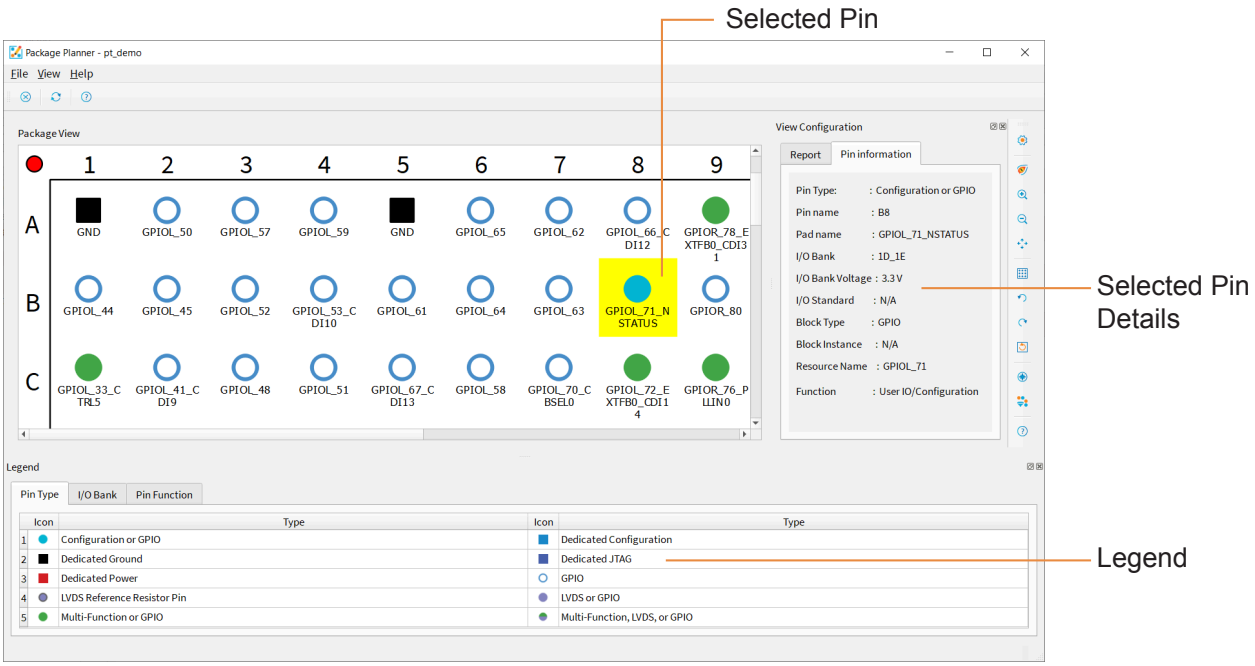
Figure 6: Package Planner



Selecting a Pin

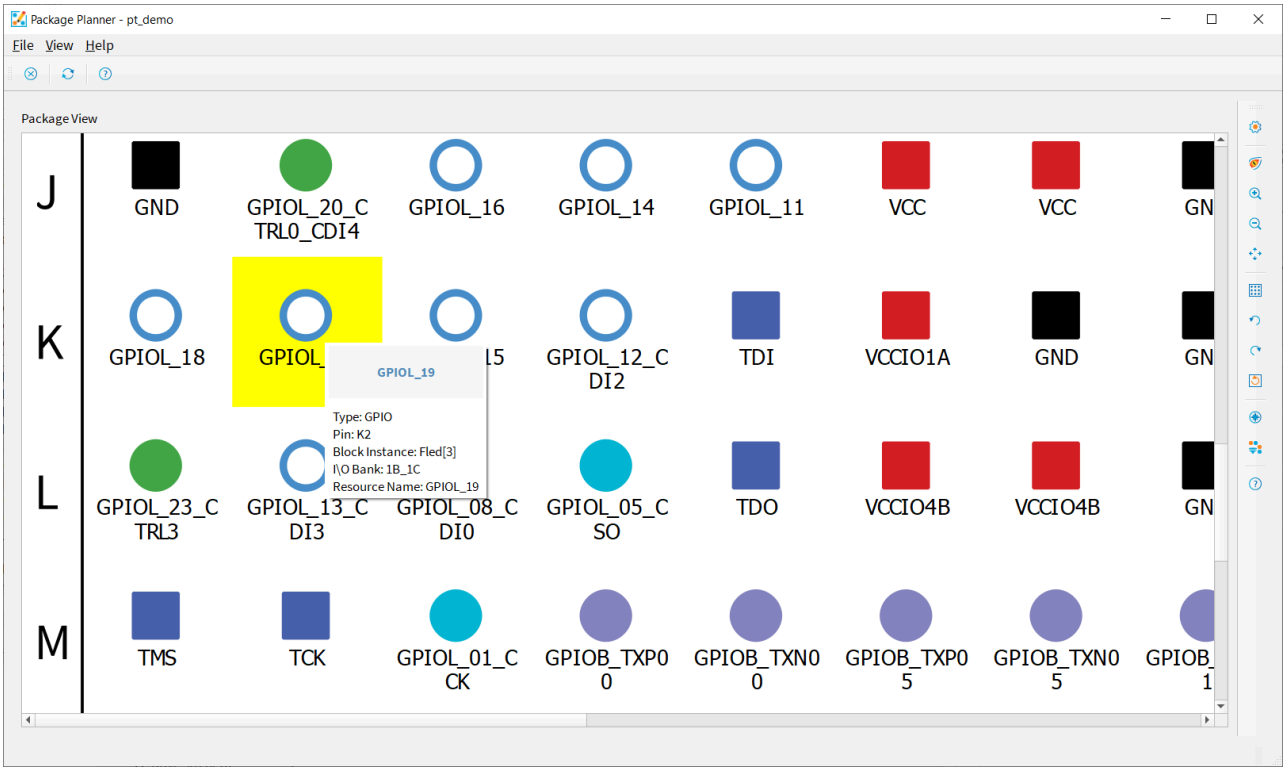
Click a pin in the pinout to highlight it. The **Pin Information** tab opens to show the details about the selected pin. Open the Legend to view the meaning of the pins' color coding.

Figure 7: Selected Pin



You can also hover over a pin for a quick view of the pin details.

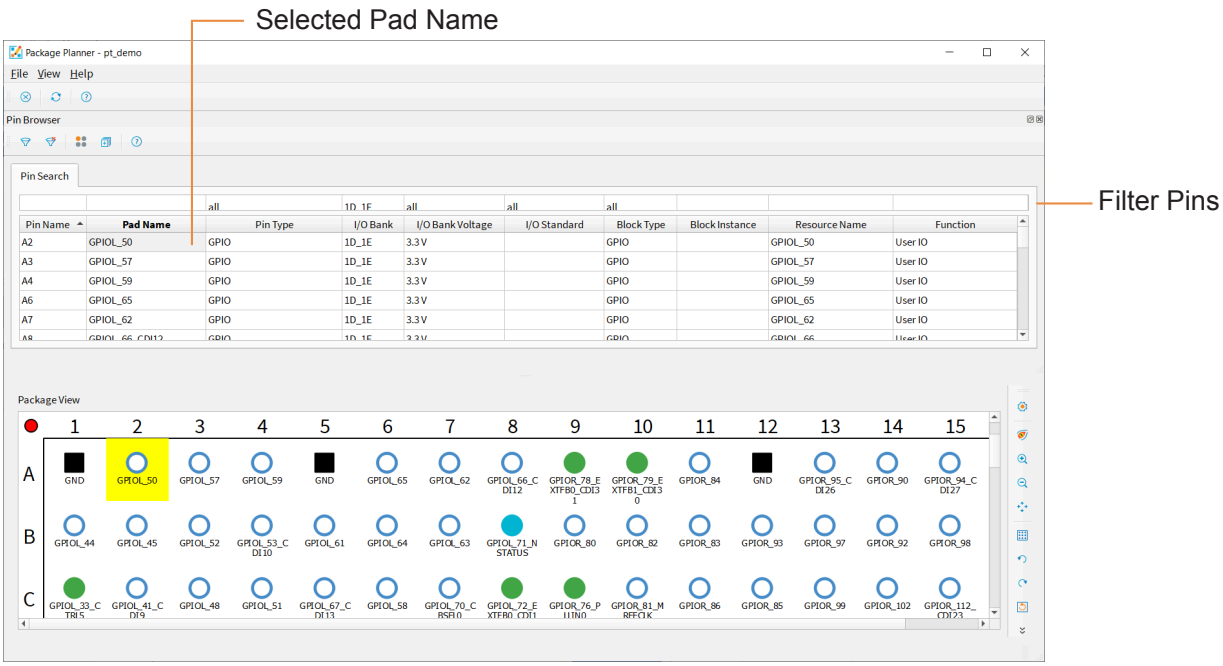
Figure 8: Pin Quick View



Browsing for Pins

The Package Planner has a **Pin Browser**, which has a table view similar to the **Resource Assigner**. You can filter pins and then select them in the **Pin Browser**. The selected pin is highlighted in the pinout.

Figure 9: Browsing for Pins



Interface Designer Output Files

When you generate constraint files, the Interface Designer creates the following output files. You can view them in the Interface section of the Result pane.

- **<project name>.interface.csv**—Constrains the FPGA design pins used in the interface between the core and the periphery.
- **<project name>.pt.rpt**—Provides information about the interface.
- **<project name>.pinout.csv**—Contains the board design pinout in CSV format.
- **<project name>.pinout.rpt**—Has the board design pinout in a nicely formatted text file format.
- **<project name>.pt_timing.rpt**—Timing report for the Titanium interface logic.
- **<project name>.pt.sdc**—Template SDC file to constrain the FPGA design pins based on the interface configuration.
- **<project name>_template.v**—Template Verilog HDL file defining the FPGA design pins based on the interface configuration.

Scripting an Interface Design

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.[5] Efinix distributes a copy of Python 3 with the Efinity® software to support point tools such as the Debugger and to allow users to write scripts to control compilation.

You use the Efinity® Interface Designer to build the peripheral portion of your design, including GPIO, LVDS, PLLs, MIPI RX and TX lanes, and other hardened blocks. Efinix provides a Python 3 API for the Interface Designer to let you write scripts to control the interface design process. For example, you may want to create a large number of GPIO, or target your design to another board, or export the interface to perform analysis. This user guide describes how to use the API and provides a function reference.

Learn more: Refer to the Python web site, **www.python.org/doc**, for detailed documentation on the language.

Learn more: For more information on using the Python API to script an interface, refer to the **Efinity Interface Designer Python API**.

[5] Source: **What Is Python? Executive Summary**

Chapter 2

Device Settings

Contents:

The Interface Designer has device-wide settings for I/O banks and configuration.

Configuration Interface

The Configuration device-wide setting lets you control or monitor configuration using the FPGA design implemented in the FPGA core.

Enable Internal Reconfiguration

Efinix® FPGAs have an internal reconfiguration feature that allows you to control reconfiguration of the FPGA from within the FPGA design. Leave this feature disabled unless you want to use internal reconfiguration.

To enable internal reconfiguration:

1. Click **Device Setting > Configuration**.
2. In the Block Editor **Remote Update** tab, turn on **Enable Internal Reconfiguration Interface**. For Ti165, Ti240, and Ti375 FPGAs, you can specify the number of retries in the Efinity software (v2024.1 and higher) by going to: **Interface Designer > Configuration > Remote Update tab > Remote Update Retries** box.
3. Indicate the name of the clock pin that will control the internal reconfiguration.
4. Define the FPGA pins that the interface uses.
5. Save.

Note: Refer to AN 010: Using the Internal Reconfiguration Feature to Update EfinixFPGAs Remotely for instructions on how to use this feature.

Table 4: Remote Update Tab Settings

Parameter	Choices	Notes
Enable Internal Reconfiguratio Interface	On, off	Default: off.
Clock Pin Name	User defined	Specify the clock pin name used to latch cfg_CBSEL when cfg_ENA is high.
Invert Clock	On, off	Default: off. Turn o to invert the clock pin.

Parameter	Choices	Notes
Image Selector [1:0] Bus Name	User defined	Multi-image select signals to the internal reconfiguration interface (not package pins). Use these signals to choose which image to load from flash memory. Efinix recommends using the default name.
Image Selector Capture Pin Name	User defined	When cfg_ENA is high, read the value of cfg_CBSEL. Efinix recommends using the default name.
Configuration Control Pin Name	User defined	Asynchronous control that initiates reconfiguration. Efinix recommends using the default name.
Error Status Pin Name	User defined	Status signal. Signal is set to 0 during power-up. Efinix recommends using the default name.
Remote Update Retries	0-7	Indicate how many times the FPGA shoud attempt to perform the remote update.

Enable User Status Pin

Titanium FPGAs have a user status pin to indicate that the FPGA has finished configuration and is in user mode. When this pin goes high, the FPGA is in user mode.

Note: This pin is only available for FPGAs that have transceivers. Refer to Package/Interface Support Matrix on page 10.

Table 5: User Status Tab Settings

Parameter	Choices	Notes
Enable User Status Control	On, off	Default: off.
User Status Pin Name	User defined	Specify the pin name. Efinix recommends using the default.

Enable External Access to Flash

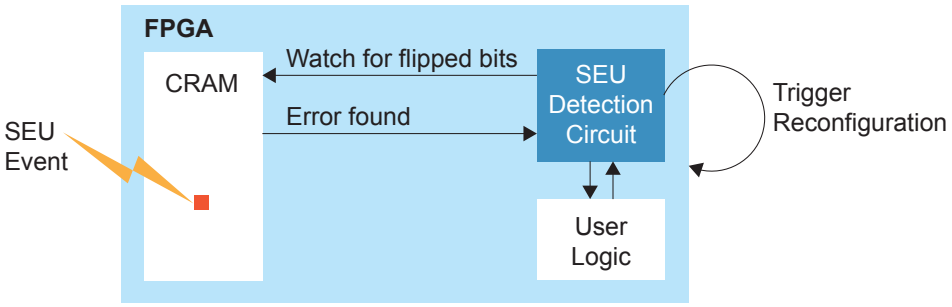
Titanium FPGAs in F100S3F2 packages have an internal SPI flash memory. If you want to write a new bitstream to the SPI flash memory while the FPGA is in user mode, turn on the **Configuration > External Flash Control > Enable external controller acccess to flash memory** option.

About SEU Detection

An SEU happens when an environmental factor, such as background radiation, causes a digital circuit to malfunction. For FPGAs, the most frequent (and most worrisome) outcome of an SEU is that a CRAM bit is changed from its programmed value. Designs may not use every CRAM bit in the FPGA, so an SEU may or may not cause the FPGA to malfunction. However, in many situations the safest course of action is to assume that the FPGA's behavior is corrupted until it is reconfigured.

Titanium FPGAs contain built-in circuitry to help detect SEUs. This circuitry periodically monitors the FPGA's CRAM, detects if a CRAM value has changed from the programmed state, and sends status signals to user logic. The user logic can optionally trigger the FPGA to reconfigure using the SEU detection circuitry.

Figure 10: SEU Detection Circuitry



Titanium FPGAs can monitor the CRAM while the FPGA is operating normally in user mode; When the SEU detection circuitry is triggered, it calculates a 32-bit CRC value based on CRAM values and compares it to a CRC computed by the Efinity software and stored in the configuration bitstream. If the values are different, the SEU circuit determines an error has occurred and sends an error signal to the user logic. You can trigger the SEU detection circuitry automatically on a set interval, or manually using a signal.

Note: For Ti35 and Ti60 FPGAs, your design should be in an "idle" state before performing an SEU check.

Enable SEU Detection

To enable the SEU feature in the Efinity software:

1. Click **Device Setting > Configuration**.
2. Click the **SEU Detection** tab.
3. Turn on **Enable SEU Detection**.
4. Choose the mode, **auto** or **manual**.
   - In **auto** mode, you can specify the amount of time between SEU error checks in microseconds. Allowable values are 1 to 1650000.0 (default).
   - In **manual** mode, you specify the pin name that controls when the SEU error check happens.

Tip: For environment that have a higher risk of SEUs, you can set a shorter wait interval. However, the shorter the wait interval, the more power the system consumes. To use less power, choose a longer wait interval. To save even more power, you can use manual mode to only trigger the SEU detection circuitry when conditions require it.

5. Save.

Note: The software issues an error if you turn on SEU detection for Ti60ES FPGAs because they do not support SEU checking.

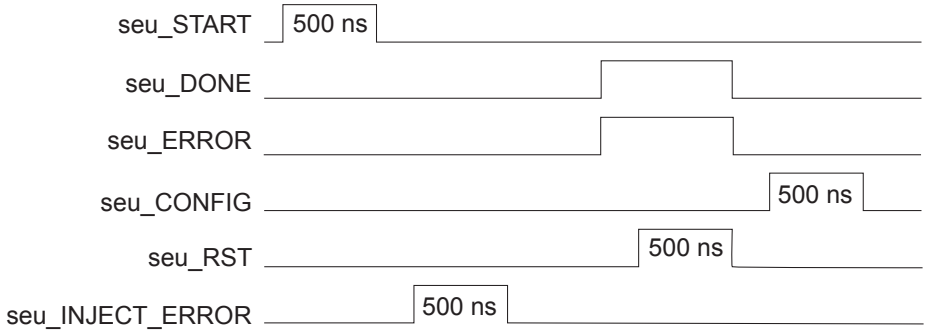
SEU Detection Circuitry

Your user logic connects to the SEU detection circuitry using the following pins.

Table 6: SEU Detection Pins

GUI Option	Default Signal Name	Direction	Description
SEU Start Detection Pin Name	seu_START	Input	Manual mode only. To use this pin to initiate an SEU check, pulse this signal high for a minimum pulse of 500 ns.
Error Injection Pin Name	seu_INJECT_ERROR	Input	This signal forces an SEU error so you can test the how the SEU detection circuitry interacts with your user design during testing and development. To inject an error, pulse this signal high for a minimum pulse of 500 ns. Pull this signal low when you are not using it.
Error Reset Pin Name	seu_RST	Input	This pin sets the Error Status pin to low to clear the error and restart SEU detection monitoring. To reset SEU monitoring, pulse this signal high for a minimum pulse of 500 ns.
Error Status Pin Name	seu_ERROR	Output	If the FPGA detects an SEU, this signal goes high. It does not go low until you toggle the Error Reset pin.
Reconfiguration Pin Name	seu_CONFIG	Input	If the FPGA is using active configuration mode, you can use this pin to trigger the FPGA to reconfigure; the FPGA does not automatically reconfigure when an error is detected. Pulse this signal high for a minimum pulse width of 500 ns. If you are using passive configuration mode or JTAG configuration, you cannot trigger reconfiguration with this signal.
SEU Done Detection Pin Name	seu_DONE	Output	This signal goes high when the SEU detection circuitry has completed calculating the CRC and comparing it to the stored one. If an SEU occurred, this signal stays high until you reset the circuitry.

Figure 11: SEU Signal Waveform



Design Check: SEU Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

seu_rule_start (error)

Message	Empty SEU Start Detection pin name found Invalid SEU Start Detection pin name found
To fix	Enter a valid start pin name.

seu_rule_interval (error)

Message	Invalid wait interval <#> microsec
To fix	Change the wait interval to one that is in range. Allowable values are 0 - 1677721.

seu_rule_empty_pins (error)

Message	Empty pin names found: <pin_type>
To fix	Enter a valid pin name.

seu_rule_invalid_pins (error)

Message	Invalid pin names found: <pin names>
To fix	Enter a valid pin name.

seu_rule_error (error)

Message	Empty Error Status pin name found Invalid Error Status pin name found
To fix	Enter a valid error pin name.

seu_rule_param (error)

Message	Invalid parameters configuration: <feature list>
To fix	One of the parameters you set was incorrect. Review any other errors for details.

seu_rule_es_device (error)

Message	SEU Detection is not supported in ES device
To fix	You get this error if you try to enable SEU for the Ti60ES. SEU is not supported in this FPGA.

Design Check: Configuration Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

configuration_rule_clock (error)

Message	Internal Reconfiguration Interface is enabled but clock pin name is invalid
To fix	Enter a valid clock name.

configuration_rule_in_user_pin (error)

Message	User Status Pin Name should be configured for PCIe
To fix	The PCI Express interfaces requires the cfg_USR_STATUS pin. Enable the pin in the Interface Designer by turning on the **Enable User Status Control** option in the **Design Explorer** > **Configuration** > **Remote Update tab**. You can use the default pin name or choose your own name.
Message	User Status Pin Name should not be empty when enable user status control
To fix	When the **Enable User Status Control** is turned on, you need to specify a name for the pin in the User Status Pin Name box (**Design Explorer** > **Configuration** > **Remote Update tab**).
Message	No PCIe is configured for user status control.
To fix	The **Enable User Status Control** option is used with the PCI Express interface. Create a new PCI Express interface block or turn off the option.

ext_flash_rule_spi_flash (error)

Message	External controller access to flash memory cannot be enabled with the usage of SPI Flash
To fix	The SPI Flash can only be accessed either by the user logic in the FPGA or an external controller outside the FPGA. Disable the external controller access if you are using the SPI Flash Interface instance or delete the SPI Flash Interface instance in the design if you want to use an external flash controller.

ext_flash_rule_res_conflict (warning)

Message	The following instances use resources that can potentially corrupt the contents of the flash memory when the external controller access to flash memory is enabled <list of instances>
To fix	Some GPIO resources are connected to the internal SPI flash in F100F3S2 packages (See **Table 124: SPI Flash Resource Assignments** on page 193). You cannot use these resources when you enable the external controller access. Disable the external flash controller, or assign different resources.

I/O Banks Interface

The I/O Banks setting shows the device I/O banks and the I/O voltage each bank uses. Some I/O banks support multiple I/O standards, and you can specify which standard the bank uses. These settings determine the FPGA pinout requirements and timing values of the interface blocks. Some I/O banks can support multiple I/O standards as long as the I/O voltages of the different standards are compatible.

To set the I/O voltage for a bank:

1. Click **Device Setting > I/O Banks**.
2. In the Block Editor, select the I/O voltage for the bank.
   You also select an I/O standard for GPIO blocks. The voltage you select for the I/O bank must be compatible with the settings you choose for any GPIO in this bank.
3. Save.

The I/O banks that contain the HVIO pins support a dynamic voltage option. You can choose between 1.8 V or one of the other voltages (2.5, 3.0, or 3.3 V). See **Dynamic Voltage Support** on page 33.

Note: The I/O banks and their legal configuration are device and package specific. Refer to the data sheet for your chosen FPGA for details on which I/O standards it supports.

Titanium I/O Banks

Efinix FPGAs have input/output (I/O) banks for general-purpose usage. Each I/O bank has independent power pins. The number and voltages supported vary by FPGA and package.

Some I/O banks are merged at the package level by sharing VCCIO pins, these are called merged banks. Merged banks have underscores (_) between banks in the VCCIO name (e.g., 1B_1C means VCCIO for bank 1B and 1C are connected). Some of the banks in a merged bank may not have available user I/Os in the package. The following table lists banks that have available user I/Os in a package.

Table 7: Titanium I/O Banks by Package for Ti35 and Ti60 FPGAs

Package	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
W64	1A, 1B, 3B	1.2, 1.35, 1.5, 1.8	-	All	1A_4B, 1B_2A, 2B_3A_3B_4A
F100	1A, 2A	1.2, 1.35, 1.5, 1.8	-	All	1A_4B, 2A_2B
	1B, 3A, 3B	1.2, 1.35, 1.5, 1.8	-	All	3B_4A
	BL	1.8, 2.5, 3.0, 3.3	✓	All	-
F100S3F2	1A, 2A	1.2, 1.35, 1.5, 1.8[6]	-	All	1A_4B, 2A_2B
	1B, 3A, 3B	1.2, 1.35, 1.5, 1.8	-	All	3B_4A
	BL	1.8, 2.5, 3.0, 3.3	✓	All	-
F225 F256	BL, TL, TR, BR,	1.8, 2.5, 3.0, 3.3	✓	All	-
	1A, 1B, 2A, 2B, 3A, 3B, 4A, 4B	1.2, 1.35, 1.5, 1.8	-	All	-

Table 8: Titanium I/O Banks by Package for Ti90 and Ti120 FPGAs

Package	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
J361	2B, 2C, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	-	All	2A_2B, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	-
J484	2B, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	-	All	2A_2B_2C, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	-
L484	2B, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	-	All	2A_2B_2C, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	-	All	-

[6] The SPI flash memory's VCC is connected to `VCCIO1A_4B`. If you are using the SPI flash memory, drive the `VCCIO1A_4B` with a 1.8 V supply.

Package	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
G400, G529	2A, 2B, 2C, 3A, 3B, 3C, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	–
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	–

Table 9: Titanium I/O Banks by Package for Ti85 and Ti135 FPGAs

Package	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
N441	2A, 2B, 2C, 4A, 4B	1.2, 1.35, 1.5, 1.8	–	All	–
	BL2, BR0, BR1, TL3, TR1	1.8, 2.5, 3.0, 3.3	✓	All	–
N484	2A, 2B, 2C, 4A, 4B	1.2, 1.35, 1.5, 1.8	–	All	–
	BL3, TR1, BR0, BR1	1.8, 2.5, 3.0, 3.3	✓	All	BR1_BL3
N676	2A, 2B, 2C, 4A, 4B	1.2, 1.35, 1.5, 1.8	–	All	–
	BL0, BL1, BL2, BL3, BR0, BR1, TL2, TL3, TR1, TR2, TR3, TR5	1.8, 2.5, 3.0, 3.3	✓	All	–

Table 10: Titanium I/O Banks by Package for Ti180 FPGAs

Package	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
J361	2B, 2C, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	2A_2B, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	–
J484, M484	2B, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	2A_2B_2C, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	–
J484D1	2A, 2B, 2C, 3A, 3B, 3C, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	2A_2B
	BL, TL, TR, BR	1.8. 2.5, 3.0, 3.3	✓	All	–
L484	2B, 3A, 3B, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	2A_2B_2C, 3B_3C
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	–
G400, G529	2A, 2B, 2C, 3A, 3B, 3C, 4A, 4B, 4C	1.2, 1.35, 1.5, 1.8	–	All	–
	BL, TL, TR, BR	1.8, 2.5, 3.0, 3.3	✓	All	–

Table 11: Titanium I/O Banks by Package for Ti165, Ti240, and Ti375 FPGAs

Package	GPIO Type	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
N484	HSIO	2B, 2C, 2D, 2E, 4B, 4C, 4D	1.2, 1.35, 1.5, 1.8	–	All	2A_2B_2C, 4A_4B

Package	GPIO Type	I/O Banks	Voltage (V)	Dynamic Voltage Support	DDIO Support	Merged Banks
	HVIO	BR0, BR3, TR1	1.8. 2.5, 3.0, 3.3	Yes	All	BR3_BR4
C529	HSIO	2A, 2B, 2C, 2D, 2E, 4B, 4C, 4D	1.2, 1.35, 1.5, 1.8	–	All	4A_4B
	HVIO	BL2, BL3, BR0, BR3, TL1, TL5, TR0, TR1, TR2	1.8. 2.5, 3.0, 3.3	Yes	All	BR3_BR4, TL1_TL5 BL2_BL3
N900	HSIO	2A, 2B, 2C, 2D, 2E, 4A, 4B, 4C, 4D	1.2, 1.35, 1.5, 1.8	–	All	–
	HVIO	BL2, BL3, BR0, BR3, BR4, TL0, TL1, TL5, TR1	1.8. 2.5, 3.0, 3.3	Yes	All	TL1_TL5
N1156	HSIO	2A, 2B, 2C, 2D, 2E, 4A, 4B, 4C, 4D	1.2, 1.35, 1.5, 1.8	–	All	–
	HVIO	BL0, BL1, BL2, BL3, BR0, BR1, BR3, BR4, TL0, TL1, TL5, TR0, TR1, TR2, TR3, TR5	1.8. 2.5, 3.0, 3.3	Yes	All	TL1_TL5

Learn more: Refer to the FPGA pinout for information on the I/O bank assignments.

Dynamic Voltage Support

Titanium HVIO I/O banks support dynamic voltage shifting. This feature lets you change the voltage to the I/O bank during user mode. There are two methods for changing the voltage:

- If you are not using 1.8 V at all, you can simply change the voltage at any time. For example, you can change from 3.3 V to 2.5 V and then back to 3.3 V.
- If you are changing the voltage to 1.8 V, you must enable an option in the Interface Designer (which creates a mode select pin <*bank name*>_MODE_SEL) and follow the timing requirements for the voltage change as described below. The value of the mode select pin chooses the voltage:
  - *High*—Changes the voltage to 1.8 V
  - *Low*—Changes the voltage back to 2.5, 3.0, or 3.3 V

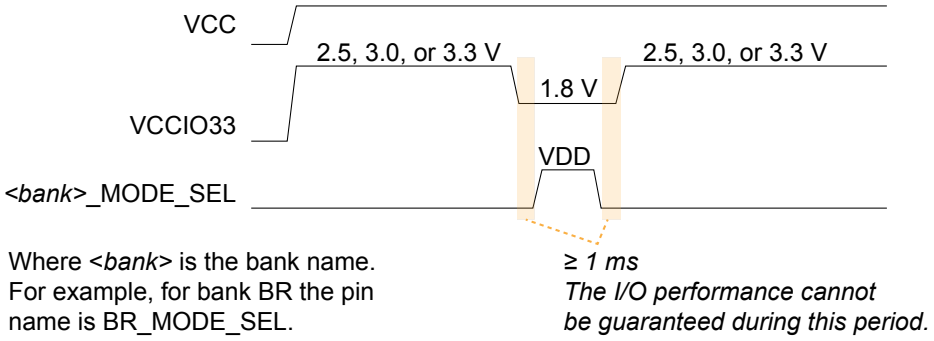
To enable the option in the Interface Designer:

1. Choose **Device Setting > I/O Banks**.
2. Turn on **Enable Dynamic Voltage**.
3. Save.
4. Add the mode select pin (<*bank*>_MODE_SEL) to your RTL design.
5. Recompile.

Note: Some HVIO I/O banks may not support the <*bank name*>_MODE_SEL pin. For example, if the I/O bank has dedicated I/O pins such as JTAG pins, then you cannot change the voltage.

When switching the voltage, pull down the voltage to 1.8 V for at least 1 ms before pulling the mode select signal high. Similarly, wait for at least 1 ms after releasing the mode select signal before changing the voltage back to the higher level.

Figure 12: VCCIO Switching Waveform (2.5, 3.0, or 3.3 V to 1.8 V)



Design Check: I/O Bank Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

io_bank_rule_dyn_voltage (error)

Message	Bank <name> does not support dynamic voltage
To fix	Choose an HVIO I/O bank that supports dynamic voltage.

io_bank_rule_lvds (error)

Message	I/O Voltage has to be set to <#> when <*LVDS, SLVS, LVDS TX, LVDS RX*> is used
To fix	Set the I/O bank voltage to the specified voltage if you are using pins in that bank for LVDS. Note: you cannot use MIPI lanes and LVDS pins in the same I/O bank because they require different voltages.

io_bank_rule_mipi_dphy (error)

Message	I/O Voltage has to be set to <#> when MIPI LANE is used
To fix	When the HSIO pins are used as MIPI lanes, the I/O bank voltage must be 1.2 V. Note: you cannot use MIPI lanes and LVDS pins in the same I/O bank because they require different voltages.

io_bank_rule_es_device (error)

Message	Unsupported 2.5 V in ES device
To fix	The Ti60ES FPGA does not support 2.5 V. Choose another voltage.

io_bank_rule_voltage_assignment (error)

Message	I/O Voltage <voltage> is not supported for the bank
To fix	Different I/O banks support different voltages. Choose a voltage that the I/O bank supports (refer to **Titanium I/O Banks** on page 30).

io_bank_rule_mode_sel (error)

Message	Empty Mode Select pin name found
To fix	Enter the name of the mode select pin.

io_bank_rule_vref (warning)

Message	It is not advisable to enable single-ended input SSTL/HSTL with LVDS Tx common mode in the following banks <banks>
To fix	An LVDS block with the **Output Differential Type** set to **custom** requires a VREF pin. A GPIO block using the single-ended SSTL or HSTL I/O standard also uses a VREF. The reference voltages for these standards is unlikely to be the same, so you should not use them in the same I/O bank. Instead, move one of the blocks to another bank.

Chapter 3

Clock and Control Networks

Contents:

The clock and control network is distributed through the FPGA to provide clocking for the core's LEs, memory, DSP blocks, I/O blocks, and control signals. The FPGA has global signals that can be used as either clocks or control signals. The global signals are balanced trees that feed the whole FPGA.

The FPGA also has regional signals that can only reach certain FPGA regions, including the top or bottom edges. The FPGA has regional networks for the core, right interface, and left interface blocks. The top and bottom interface blocks have 1 regional clock network each. You can drive the right and left sides of each region independently. Each region also has a local network of clock signals that can only be used in that region.

The core's global buffer (GBUF) blocks drive the global and regional networks. Signals from the core and interface can drive the GBUF blocks.

Each network has dedicated enable logic to save power by disabling the clock tree. The logic dynamically enables/disables the network and guarantees no glitches at the output.

Figure 13: Global and Regional Clock Network Overview (Ti35, Ti60)

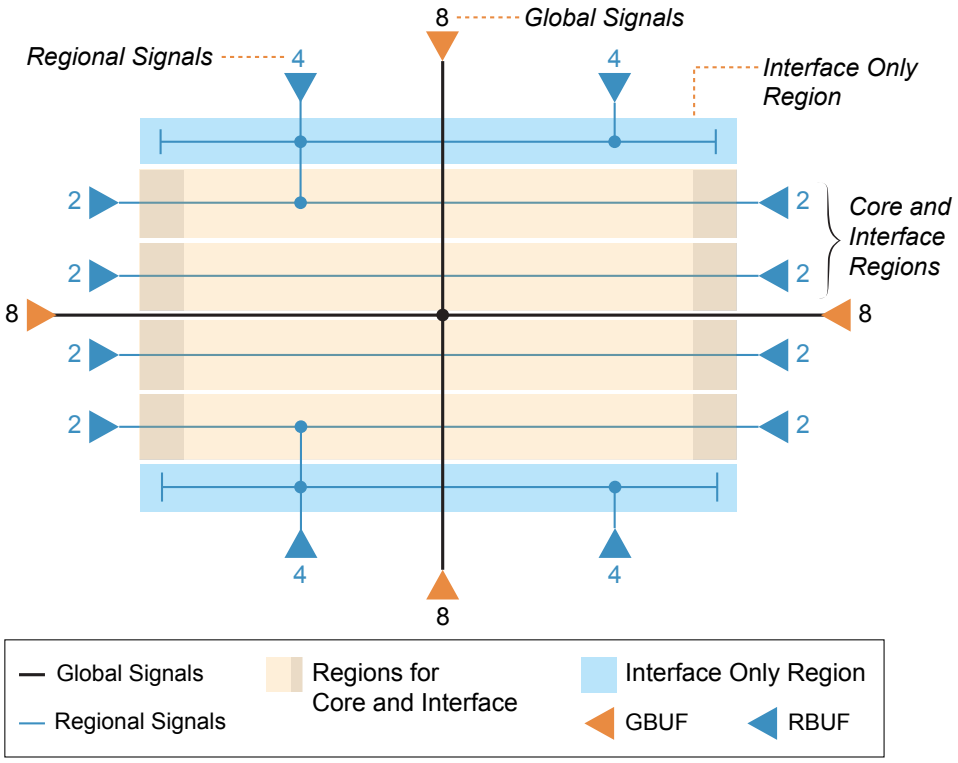


Figure 14: Global and Regional Clock Network Overview (Ti90, Ti120, Ti180)

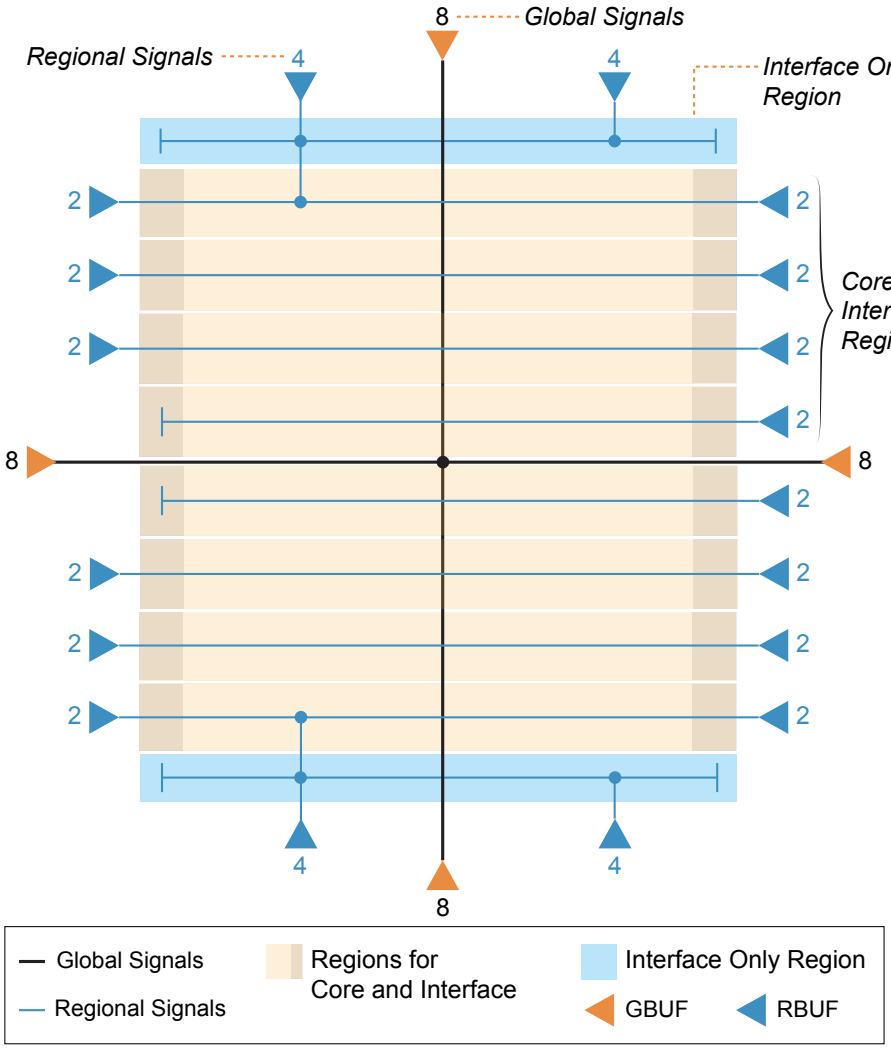


Figure 15: Global and Regional Clock Network Overview (Ti85, Ti135)

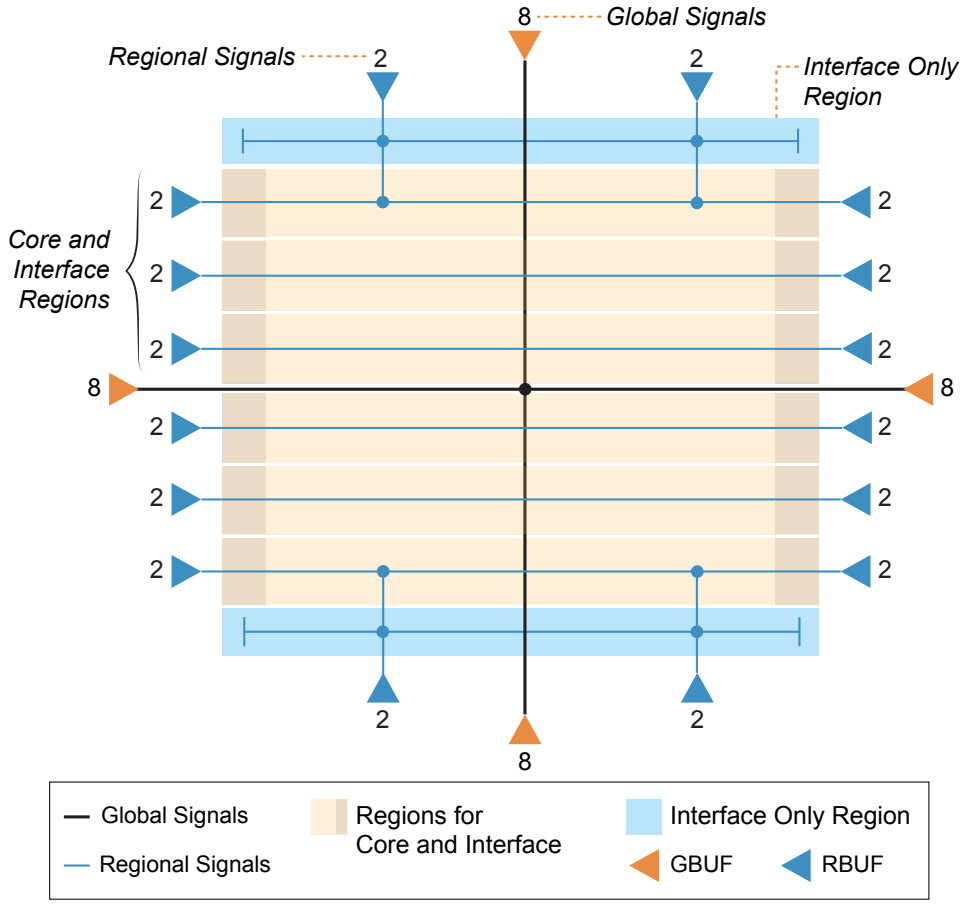
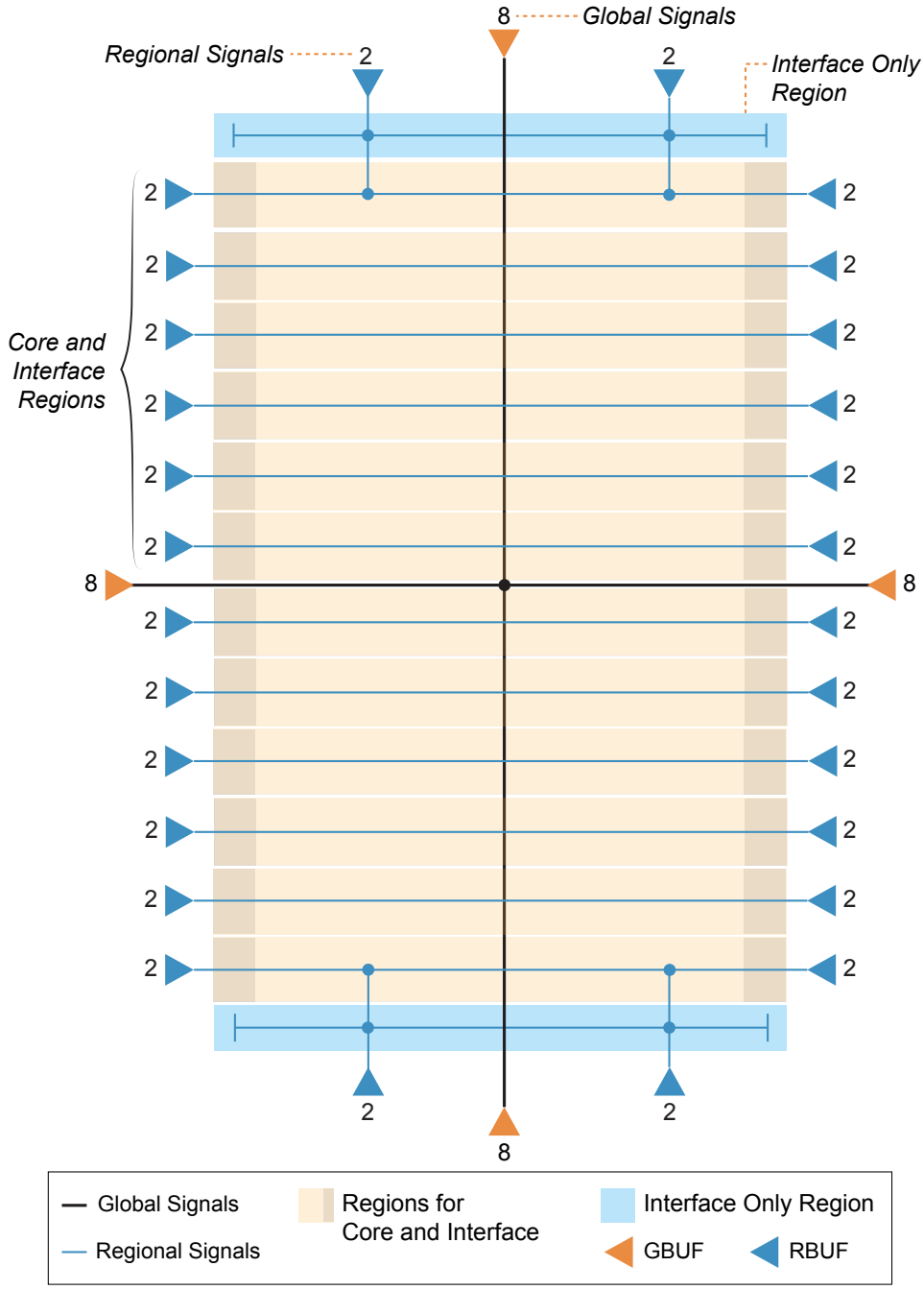


Figure 16: Global and Regional Clock Network Overview (Ti165, Ti240, Ti375)



Note: For detailed descriptions of the clock networks, refer to the data sheet.

Clock Sources that Drive the Global and Regional Networks

The Titanium global and regional networks are highly flexible and configurable. Clock sources can come from interface blocks, such as GPIO or PLLs, or from the core fabric.

Note: For more information on the clock sources that can drive the global and regional networks, refer to the data sheet.

Table 12: Clock Sources that Drive the Global and Regional Networks

Source	Description
GPIO	Supports GCLK and RCLK. (Only the P resources support this connection type).
LVDS RX	Supports GCLK and RCLK.
MIPI D-PHY RX, TX, and SSC PLL	**Ti85, Ti135, Ti90, Ti120, Ti165, Ti180, Ti240, Ti375:** Can drive the word clock onto the global and regional clock networks.
Transceiver TX, RX, and PIPE P clocks	**Ti85, Ti135, Ti165, Ti240, Ti375:** Can drive the global and regional clock networks on the right side.
MIPI RX Lane (configured as clock lane)	Supports GCLK (default) and RCLK. You can only use resources that are identified as clocks.
PLL	Output clocks 0 - 3 connect to the global network. Output clock 4 only connects to the regional network in the top or bottom interface regions (depending on the location of the PLL) and can only drive interface blocks on the top or bottom of the FPGA. **Ti35, Ti60:** Output clocks 0 - 3 connect to the global network. Output clock 4 only connects to the regional network in the top or bottom interface regions (depending on the location of the PLL) and can only drive interface blocks on the top or bottom of the FPGA. **Ti85, Ti135, Ti90, Ti120, Ti165, Ti180, Ti240, Ti375:** All output clocks connect to the global network. Some PLLs can also drive the regional clock network; see "Driving the Regional Network" in the data sheet for details.
Oscillator	Connects to global buffer.[7]
Core	Signals from the core logic can drive the global or regional network.

Note: Some clock sources can drive both the global and regional networks. See **Driving both the Global and Regional Networks** on page 40 for instructions on using both.

[7] The Ti60ES oscillator can only drive the core.

Configuring the Dynamic Clock Multiplexers

You configure the dynamic clock multiplexers in the Interface Designer. Expand **Device Setting > Clock/Control Configuration** and then click the multiplexer for the bottom, left, right, or top.

Table 13: Global Buffer Configuration

Option	Choices	Description
Core Clock *n* Pin Name	User defined	Specify a clock from the core that drives the global buffer. You can use up to four.
Enable Dynamic Mux 0 Enable Dynamic Mux 7	On or off	Turn on this option to enable the dynamic mulplexer. The clock multiplexers (static and dynamic) are numbered from 0 to 7. The dynamic ones are 0 and 7.
Dynamic Clock Mux Select [1:0] Bus Name	User defined	Specify the bus that controls the dynamic multiplexer.
Dynamic Clock Pin Name	User defined	Specify the clock name that drives the core RTL design.
Dynamic Clock Input n	List of sources[8] or None	Choose the clock source. Unassigned indicates that you have not yet used that resource. Choose **None** if you want to leave an input unassigned. A clock source must be assigned to the dynamic clock multiplexer input 0.
Independently Connect to Core	On or off	Enable this option if you want the clock source to also be available to the core.

Driving both the Global and Regional Networks

In some situations you may want a clock source to drive both the global and regional clock networks. In this case, you use the **rclk** connection type. To drive both a regional and a global, make these settings in the Interface Designer:

1. Create your clock source (e.g., a GPIO).
2. For the **Connection Type**, choose **rclk**.
3. In the Resource Assigner, assign a resource that has RCLK shown as an alternate connection.
4. Note the letter after GPIO in the resource name. This letter indicates the side of the FPGA. GPIO*B* = bottom, GPIO*T* = top, etc.
5. Click **Device Setting > Clock/Control Configuration**.
6. Select the bottom, left, right, or top as determined in step 4.
7. Click the **Regional Buffers** tab.
8. Choose the regional buffer according to "Driving the Regional Network" in the data sheet. When you select a buffer, the resource for that buffer displays under **Assignment**.
9. Specify a **Global Pin Name** to drive the global network.

[8] Refer to the Driving the Global Network topic in the data sheet for the list of available clock sources.

Design Check: Clock Control Messages

When you check your design, the Interface Designer applies design rules to your clock and control settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

clock_rule_capacity (error)

Message	Cannot connect to more than <int> different clocks per region (40 rows) on left and right and <int> clocks on the top or bottom
To fix	You cannot have more than 32 clocks (GPIO configured in clkout mode) coming from the core. You need to remove some clocks.

Message	Cannot connect to more than <int> different clocks per region (40 rows) on left and right
To fix	You are using more clocks that are available for the local region. Remove some clocks. See "Driving the Local Network" in the data sheet.

Message	Cannot connect to more than <int> different clocks on top and bottom
To fix	You are using more clocks than are available on the top and bottom local regions. Remove some clocks. See "Driving the Local Network" in the data sheet.

clock_rule_max_count (error)

Message	Number of core clock used exceeds max limit of <int>
To fix	You cannot have more than 32 clocks (GPIO configured in clkout mode) coming from the core. You need to remove some clocks.

clock_rule_lvds_bidir_fast_clk_comm (warning)

Message	Some serial clocks used in certain bank instances have a propagation delay that differs from other banks, thereby impacting the subsequent clock: [<fast clock name> (Different propagation bank: <bank_name> Instances: <list of instance name>)]
To fix	You get this warning if you use the same serial clock for LVDS Bidir instances that use resources from different banks. Update the instance's resource to use resources from the same bank mentioned in the message.

clock_rule_lvds_rx_clock_source (warning)

Message	The following PLL instance has output clocks driving LVDS Rx instance on different sides pair - [<PLL instance>: [<clock name>(<LVDS instance>)]]
To fix	The PLL output clocks go to multiple LVDS RX on different sides of the FPGA. The fast clock and slow clock can only drive I/O from the same left/right or top/bottom sides. For example, TR_PLL generates rx_fastclk and rx_slowclk for LVDS in the right-side bank. These clocks can also drive the LVDS channel on the left bank. However, they cannot drive LVDS in the top or bottom banks.

clock_rule_lvds_rx_fast_clk_comm (warning)

Message	Some serial clocks used in certain bank instances have a propagation delay that differs from other banks, thereby impacting the subsequent clock: [<fast clock name> (Different propagation bank: <bank_name> Instances: <list of instance name>)]
To fix	You get this warning if you use the same serial clock for LVDS RX instances that use resources from different banks. Update the instance's resource to use resources from the same bank mentioned in the message.

clock_rule_lvds_tx_fast_clk_comm (warning)

Message	Some serial clocks used in certain bank instances have a propagation delay that differs from other banks, thereby impacting the subsequent clock: [<fast clock name> (Different propagation bank: <bank_name> Instances: <list of instance name>)]
To fix	You get this warning if you use the same serial clock for LVDS TX instances that use resources from different banks. Update the instance's resource to use resources from the same bank mentioned in the message.

clock_rule_mipi_tx_fast_clk_comm (warning)

Message	Some serial clocks used in certain bank instances have a propagation delay that differs from other banks, thereby impacting the subsequent clock: [<fast clock name> (Different propagation bank: <bank_name> Instances: <list of instance name>)]
To fix	You get this warning if you use the same serial clock for MIPI TX instances that use resources from different banks. Update the instance's resource to use resources from the same bank mentioned in the message.

clock_rule_pll_ref_clock_lvds_rx (error)

Message	The following PLL instance has reference clock that does not match the side of the LVDS Rx instance driven by its output clocks - [<PLL instance>: [<clock name>(<LVDS instance>)]]
To fix	Choose the PLL reference clock that is on the same side as the LVDS that the output clock is driving. For example, if TR_PLL is driving LVDS on the right side, the PLL external source clock should also come from the I/O on the right side.

clock_rule_undefined_name (info)

Message	Clock <*clock name*> not defined in the interface, assuming core generated.
To fix	All clocks in the periphery must be defined in the Interface Designer (GPIO clock, oscillator, PLL, LVDS GCLK, MIPI D-PHY CLK). This info message indicates that you have not defined it as an interface block. If the clock is generated in the core you can ignore this message.

clkmux_rule_clocks_routed (error)

Message	Unrouted pins driving inputs of clock mux <ins name>:<inputs not routeable> Some inputs of clock mux <ins name> were not routed
To fix	The software tries to route all of the clocks according to the scheme shown in "Driving the Global Network" in the data sheet. If it cannot find a mapping, it issues this error. Reassign the instances to other resources or try using a different PLL output clock (if they are not all assigned).

clkmux_rule_core_clock_pin (error)

Message	Core clock pin can only be used with dynamic mux enabled
To fix	Clocks from the core can only drive the dynamic clock multiplexers (see "Driving the Global Network" in the data sheet). Enable a dynamic mux (0 or 7) and assign the core clock to it (**Configuring the Dynamic Clock Multiplexers** on page 40).

clkmux_rule_core_clock_static_mux (error)

Message	Core clock pin <name> not allowed to route through static mux output
To fix	Clocks that come from the core can only connect to dynamic multiplexer input (see "Driving the Global Network" in the data sheet). You need to add the clock to a dynamic mux.

clkmux_rule_dynamic_clock_pin (error)

Message	Dynamic clock pin names for <both dynamic muxes/dynamic mux 0/7> <are/is> empty
To fix	You need to specify the pin name. Go to **Device Setting** > **Clock/Control Configuration** > **<region>** > **Global Buffers tab**.

clkmux_rule_dynamic_clock_select_pin (error)

Message	Dynamic clock select pin names for <both dynamic muxes/dynamic mux 0/7> <are/is> empty
To fix	You need to specify the pin name. Go to **Device Setting** > **Clock/Control Configuration** > **<region>** > **Global Buffers tab**.

clkmux_rule_global_regional_pin (error)

Message	Missing global pin name for a regional connection that also connects to global buffer
To fix	If you want to use a clock source to drive *both* the global and regional networks, you need to specify the name of the clock that drives the global network. See **Driving both the Global and Regional Networks** on page 40.

Message	Global and regional buffer connections require unique pin name
To fix	If you are using both the global and regional buffers, you need to specify unique pin names for each one.

clkmux_rule_global_regional_resource (error)

Message	Regional buffer resource <name> does not support global connection
To fix	Some clock sources cannot connect to the global network, e.g., PLL CLKOUT4. Look in the Resource Assigner **Alt Conn** column to find a different resource that can connect.

clkmux_rule_pll_clock (warning)

Message	Dynamic clock mux <ClockMux Name> connected to both inverting and non-inverting clock sources: Clock inversion will not be applied to <Clock Names>
To fix	Disable the clock inversion option in the PLL instance properties for clocks that are connect to the dynamic clock mux.

clkmux_rule_pll_output_clock (error)

Message	Found the following PLL output clock routed multiple times: <*PLL output*>
To fix	PLL clocks connect to clock muxes on two sides of the device. Only one of these connections may be used at a time. Make sure you have not chosen a PLL clock as a dynamic clock source and enabled it to independently drive the core on two different clock muxes.

clkmux_rule_pll_serial_parallel_clocks (error)

Message	Clock <*PLL Resource.CLKOUT#*> is not found to be routed on the same side with other PLL clocks: <*list of PLL Resources.CLKOUT# that must be on the same side*>
To fix	The software tries to route all of the clocks according to the scheme shown in "Driving the Global Network" in the data sheet. If it cannot find a mapping, it issues this error. Reassign the instances to other resources or try using a different PLL output clock (if they are not all assigned).

clkmux_rule_regional_conn_type (error)

Message	Regional buffer instance <name> requires connection type to be set to rclk
To fix	If you are a clock source to drive the reginal network, you need to choose **rclk** as the **Connection Type** in the **Input** tab. (see **Input Mode** on page 72)

clkmux_rule_type_config (error)

Message	Resource <*name*> configured as MIPI LANE Rx not allowed to connect to dynamic mux <0/7> Resource <*name*> configured as GPIO/LVDS not allowed to connect to dynamic mux <0/7> Invalid output clock configuration of <name> connected to index <int> of dynamic mux <0/7> Assigned core clock pin name at index <int> of dynamic mux <0/7> is empty Resource <name> assigned to input <int> of dynamic mux <0/7> but no valid configured instance found
To fix	You get this error if you have not configured the clock source correctly to connect to the dynamic mux. For example, you can configure the same resource as GPIO, LVDS, or a MIPI lane. If you configure it as a MIPI lane, then you cannot connect it to the dynamic mux. Only GPIO or LVDS can connect. See "Driving the Global Network" in the data sheet for the sources that can drive the dynamic multiplexer.

Message	Resource <*name*> with instance <*name*> assigned to RBUF# has invalid configuration Resource <*name*> with instance <*name*> can only connect to regional buffer input # through output clock 4
To fix	You can only connect the PLL output to regional buffers on the top and bottom. Refer to "Driving the Regional Network" in the data sheet.

pma_clkmux_rule_clocks_routed (error)

Message	Unrouted pins driving inputs of pma clock mux <*instance name*>:<*inputs not routeable*>. Please refer to the summary report and device datasheet Clock Network section for more details.
To fix	Change the transceiver-related instances to other resources or use an **rclk** connection type for the clock.

Message	Some inputs of pma clock mux <*instance name*> were not routed
To fix	The software tries to route all of the clocks according to the scheme shown in "Driving the Global Network" in the data sheet. If it cannot find a mapping, it issues this error. Reassign the instances to other resources or try using a different PLL output clock (if they are not all assigned).

Chapter 4

DDR Interface

Contents:

Some Titanium FPGAs have a hardened IP interface block to communicate with off-the-shelf memories. Refer to the Package/Interface Support Matrix on page 10 to find out if your FPGA supports DDR.

About the DDR DRAM Interface

Important: Ti85, Ti135, Ti165, Ti240, Ti375: All information is preliminary and pending definition.

The DDR PHY interface supports LPDDR4/4X memories with x16 and x32 DQ widths and a memory controller hard IP block. The memory controller provides two full-duplex AXI4 buses to communicate with the FPGA core.

Note: The DDR PHY and controller are hard blocks; you cannot bypass the DDR DRAM memory controller to access the PHY directly for non-DDR memory controller applications.

Figure 17: DDR DRAM Block Diagram

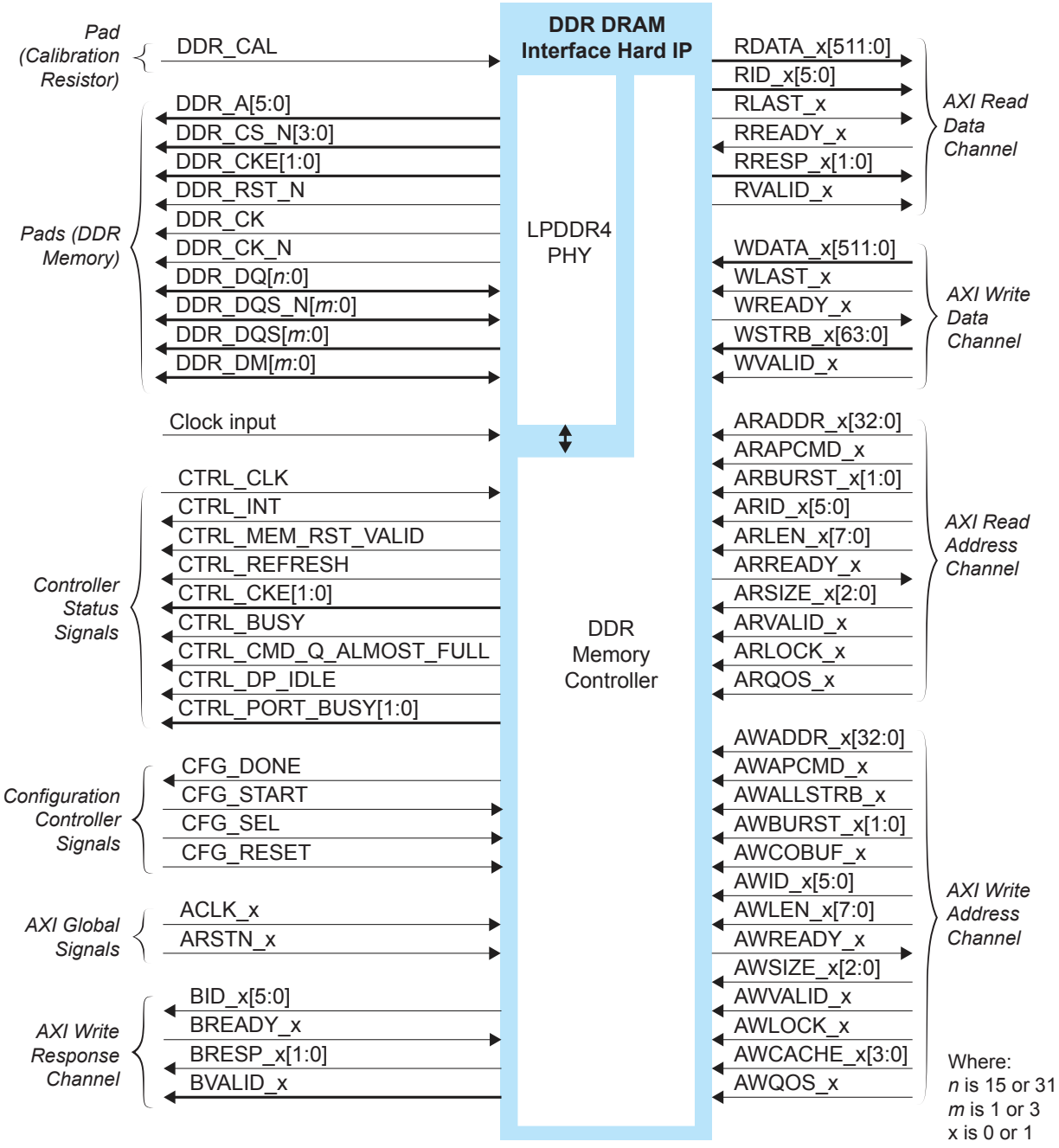
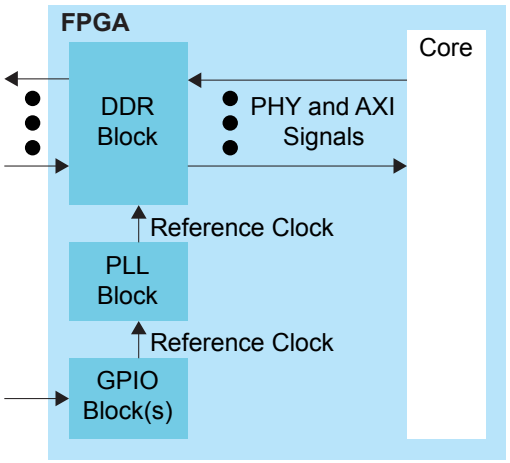


Figure 18: DDR DRAM Interface Block Diagram



Note: The PLL reference clock must be driven by I/O pads. The Efinity software issues a warning if you do not connect the reference clock to an I/O pad. (Using the clock tree may induce additional jitter and degrade the DDR performance.)

Additionally, the PLL that clocks the DDR DRAM interface should not use programmable duty cycle, fractional output, or spread-spectrum clocking because these features increase jitter.

Refer to About the PLL Interface on page 146 for more information about the PLL block.

Table 14: DDR DRAM Pads

Signal	Direction	Description
DDR_A[5:0]	Output	Address signals to the DRAM.
DDR_CS_N[3:0]	Output	Chip select to the DRAM.
DDR_CKE[1:0]	Output	Active-high clock enable signals to the DRAM.
DDR_RST_N	Output	Active-low reset signal to the DRAM.
DDR_CK	Output	Differential clock signals to the DRAM.
DDR_CK_N	Output	
DDR_DQ[*n*:0]	Bidirectional	Data bus to/from the memories. For writes, the FPGA drives these signals. For reads, the memory drives these signals. These signals are connected to the DQ pins on the memories. *n* is 15 or 31 depending on the Data Width setting. If unused, can be left floating on the board.
DDR_DQS_N[*m*:0]	Bidirectional	Differential data strobes to/from the memories. For writes, the FPGA drives these signals. For reads, the memory drives these signals. These signals are connected to the DQS pins on the memories. *m* is 1or 3 depending on the DQ width. If unused, can be left floating on the board.
DDR_DQS[*m*:0]	Bidirectional	
DDR_DM[*m*:0]	Bidirectional	Signals used as active-high data-mask and data bus inversion indicator. *m* is 1 or 3 depending on the DQ width. If data bus inversion is enabled for a write operation, the DDR controller will drive the signal high if the write data byte is inverted. Similarly, if data bus inversion is enabled for a read operation, the memory device will drive the signal high if the read data byte is inverted. If unused, can be left floating on the board.

Table 15: Calibration Resistor Pad

Signal	Direction	Description
DDR_CAL	Input	Calibration resistor connection. Connect to the ground through a 240 Ω resistor on your board.

Table 16: Controller Status Signals

Signal	Direction	Clock Domain	Description
CTRL_CLK	Input	N/A	Clock for controller status signals.
CTRL_INT	Output	N/A	Controller detects Interrupt.
CTRL_MEM_RST_VALID	Output	N/A	Controller has been reset.
CTRL_REFRESH	Output	CTRL_CLK	Indicate controller is executing refresh command.
CTRL_CKE[1:0]	Output	CTRL_CLK	Delayed 'control_cke' from the controller, indicating that the memory is in self-refresh or power down mode.
CTRL_BUSY	Output	CTRL_CLK	Controller is busy reading data.
CTRL_CMD_Q_ALMOST_FULL	Output	CTRL_CLK	Command queue reached 'q_fullness' parameter.
CTRL_DP_IDLE	Output	CTRL_CLK	Datapath is idle.
CTRL_PORT_BUSY[1:0]	Output	CTRL_CLK	Indicate if port is reading data.

Table 17: Configuration Controller Signals

Signal	Direction	Description
CFG_RESET	Input	Active-high configuration controller reset. Asserting this signal also resets the DDR controller, PHY and the DRAM device.
CFG_START	Input	Start the configuration controller.
CFG_DONE	Output	Indicates the configuration controller is done
CFG_SEL	Input	Tie this input to low to enable the configuration controller.

Table 18: AXI4 Global Signals (Interface to FPGA Core Logic)

Signal	Direction	Clock Domain	Description
ACLK_*x*	Input	N/A	AXI4 clock inputs.
ARSTN_*x*	Input	ACLK_x	Active-low reset signal to the AXI interface.

Table 19: AXI4 Write Response Channel Signals (Interface to FPGA Core Logic)

Signal x is 0 or 1	Direction	Clock Domain	Description
BID_*x*[5:0]	Output	ACLK_x	Response ID tag. This signal is the ID tag of the write response.
BREADY_*x*	Input	ACLK_x	Response ready. This signal indicates that the master can accept a write response.
BRESP_*x*[1:0]	Output	ACLK_x	Read response. This signal indicates the status of the read transfer.
BVALID_*x*	Output	ACLK_x	Write response valid. This signal indicates that the channel is signaling a valid write response.

Table 20: AXI4 Read Data Channel Signals (Interface to FPGA Core Logic)

Signal x is 0 or 1	Direction	Clock Domain	Description
RDATA_*x*[511:0]	Output	ACLK_x	Read data.
RID_*x*[5:0]	Output	ACLK_x	Read ID tag. This signal is the identification tag for the read data group of signals generated by the slave.
RLAST_*x*	Output	ACLK_x	Read last. This signal indicates the last transfer in a read burst.
RREADY_*x*	Input	ACLK_x	Read ready. This signal indicates that the master can accept the read data and response information.
RRESP_*x*[1:0]	Output	ACLK_x	Read response. This signal indicates the status of the read transfer.
RVALID_*x*	Output	ACLK_x	Read valid. This signal indicates that the channel is signaling the required read data.

Table 21: AXI4 Write Data Channel Signals (Interface to FPGA Core Logic)

Signal x is 0 or 1	Direction	Clock Domain	Description
WDATA_*x*[511:0]	Input	ACLK_x	Write data.
WLAST_*x*	Input	ACLK_x	Write last. This signal indicates the last transfer in a write burst.
WREADY_*x*	Output	ACLK_x	Write ready. This signal indicates that the slave can accept the write data.
WSTRB_*x*[63:0]	Input	ACLK_x	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
WVALID_*x*	Input	ACLK_x	Write valid. This signal indicates that valid write data and strobes are available.

Table 22: AXI4 Read Address Signals (Interface to FPGA Core Logic)

Signal x is 0 or 1	Direction	Clock Domain	Description
ARADDR_*x*[32:0]	Input	ACLK_x	Read address. It gives the address of the first transfer in a burst transaction.
ARBURST_*x*[1:0]	Input	ACLK_x	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated. 'b01 = INCR 'b10 = WRAP
ARID_*x*[5:0]	Input	ACLK_x	Address ID. This signal identifies the group of address signals.
ARLEN_*x*[7:0]	Input	ACLK_x	Burst length. This signal indicates the number of transfers in a burst.
ARREADY_*x*	Output	ACLK_x	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
ARSIZE_*x*[2:0]	Input	ACLK_x	Burst size. This signal indicates the size of each transfer in the burst.
ARVALID_*x*	Input	ACLK_x	Address valid. This signal indicates that the channel is signaling valid address and control information.
ARLOCK_*x*	Input	ACLK_x	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
ARAPCMD_*x*	Input	ACLK_x	Read auto-precharge.
ARQOS_*x*	Input	ACLK_x	QoS identifier for read transaction.

Table 23: AXI4 Write Address Signals (Interface to FPGA Core Logic)

Signal x is 0 or 1	Direction	Clock Domain	Description
AWADDR_*x*[32:0]	Input	ACLK_x	Write address. It gives the address of the first transfer in a burst transaction.
AWBURST_*x*[1:0]	Input	ACLK_x	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated.
AWID_*x*[5:0]	Input	ACLK_x	Address ID. This signal identifies the group of address signals.
AWLEN_*x*[7:0]	Input	ACLK_x	Burst length. This signal indicates the number of transfers in a burst.
AWREADY_*x*	Output	ACLK_x	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
AWSIZE_*x*[2:0]	Input	ACLK_x	Burst size. This signal indicates the size of each transfer in the burst.
AWVALID_*x*	Input	ACLK_x	Address valid. This signal indicates that the channel is signaling valid address and control information.
AWLOCK_*x*	Input	ACLK_x	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
AWAPCMD_*x*	Input	ACLK_x	Write auto-precharge.
AWQOS_*x*	Input	ACLK_x	QoS identifier for write transaction.
AWCACHE_*x*[3:0]	Input	ACLK_x	Memory type. This signal indicates how transactions are required to progress through a system.
AWALLSTRB_*x*	Input	ACLK_x	Write all strobes asserted. The DDR controller only supports a maximum of 16 AXI beats for write commands using this signal.
AWCOBUF_*x*	Input	ACLK_x	Write coherent bufferable selection.

Note: Refer to the data sheet for information on which PLL resources can provide the DDR DRAM interface input clocks.

Using the DDR Interface

The following tables describe the settings for the Titanium DDR block in the Interface Designer. (See which packages support DDR.)

Table 24: Base Tab

Parameter	Choices	Notes
Instance Name	User defined	Indicate the DDR instance name. This name is the prefix for all DDR signals.
DDR Resource	None, DDR_0, DDR_1	Indicate the resources available.
Data Width	16, 32	Choose the DQ width. Default: 32 (16 for J361 and M484 packages)
Memory Density	2G, 3G, 4G, 6G, 8G, 12G, 16G	Choose the memory density per channel. Default: 4G
Physical Rank	1, 2	Default: 1
Memory Type	LPDDR4/4X	Default: LPDDR4
Clock	CLKIN 0, CLKIN 1, CLKIN 2	Choose which PLL resource to use as the DDR clock. **CLKIN 0**–PLL_TL0 **CLKIN 1**–PLL_TL1 **CLKIN 2**–PLL_TL2 (default)

Table 25: Advanced Options Tab (FPGA Settings)

Perform IBIS simulation to determine which settings are the most suitable for your application. IBIS models are available in the Efinix Support Center.

Option	Choices	Notes
DQ Pull-Down Drive Strength (Ohm)	34.3, 40, 48, 60, 80, 120, 240	Default: 48
DQ Pull-Down ODT (Ohm)	34.3, 40, 48, 60, 80, 120, 240, High-Z	Default: 60
DQ Pull-Up Drive Strength (Ohm)	34.3, 40, 48, 60, 80, 120, 240	Default: 48
DQ Pull-Up ODT (Ohm)	34.3, 40, 48, 60, 80, 120, 240, High-Z	Default: High-Z
VREF Range Selection	Range 0, Range 1	Default: Range 0
VREF Setting (% of VDDQ)	**LPDDR4, Range 0:** 5.40 - 38.42 (step: 0.26) **LPDDR4, Range 1:** 11.90 - 48.222 (step: 0.286) **LPDDR4x, Range 0:** 11.60 - 49.70 (step: 0.3) **LPDDR4x, Range 1:** 21.20 - 59.30 (step: 0.3)	Default: LPDDR4, Range 0: 21.78 LPDDR4, Range 1: 29.918 LPDDR4x, Range 0: 30.5 LPDDR4x, Range 1: 40.1

Table 26: Advanced Options Tab (Memory Mode Register Settings)

Option	Choices	Notes
Burst Length	BL = 16 Sequential, BL = 16 or 32 Sequential, BL = 32 Sequential	Default: 16 Sequential
CA Bus Receiver On-Die-Termination for CS0/CS1	Disable, RZQ/1, RZQ/2, RZQ/3, RZQ/4, RZQ/5, RZQ/6	Default: Disable
DQ Bus Receiver On-Die-Termination for CS0/CS1	Disable, RZQ/1, RZQ/2, RZQ/3, RZQ/4, RZQ/5, RZQ/6	Default: Disable
Pull-Down Drive Strength (PDDS) for CS0/CS1	RFU, RZQ/1, RZQ/2, RZQ/3, RZQ/4, RZQ/5, RZQ/6	Default: RZQ/6
CA VREF Setting Range Selection	RANGE [0], RANGE [1]	Default: RANGE [1]
CA VREF Settings (% of VDD2)	**RANGE [0]:** 10 - 30 (step: 0.4) **RANGE [1]:** 22 - 42 (step: 0.4)	Default: RANGE [0]: 27.2 RANGE [1]: 27.2
DQ VREF Setting Range Selection	RANGE [0], RANGE [1]	Default: RANGE [1]
DQ VREF Settings (% of VDDQ)	**RANGE [0]:** 10 - 30 (step: 0.4) **RANGE [1]:** 22 - 42 (step: 0.4)	Default: RANGE [0]: 27.2 RANGE [1]: 27.2
Enable DBI Write	On or off	Enable data bus inversion (DBI) for write operation. When enabled, the controller automatically inverts the write data byte if there are 5 or more '1's in the data bus, and indicates the data is inverted by driving the DDR_DM signal high. Default: On
Enable DBI Read	On or off	Enable data bus inversion (DBI) for read operation. When enabled, the controller automatically inverts the read data byte if the DDR_DM signal is high. Default: On
CK ODT CS0/CS1 Enabled for Non-terminating Rank	Override Disabled, Override Enabled	Default: Override Disabled
CS ODT CS/CS1 Enabled for Non-terminating Rank	Override Disabled, Override Enabled	Default: Override Disabled
CA ODT CS/CS1 Termination Disable	Obeys ODT_CA Bond Pad, Disabled	Default: Override Obeys ODT_CA Bond Pad

Table 27: Config Controller Tab

Option	Choices	Notes
<description> Pin Name	User defined	Configuration and control pins. You can use the default names or specify your own.

Table 28: Advanced Options Tab (Memory Timing Settings)

Option	Choices	Notes
tCCD, CAS-to-CAS Delay (cycles)	8 - 31	Default: 8
tCCDMW, CAS-to-CAS Delay Masked Write (cycles)	32 - 63	Default: 32
tFAW, Four-Bank Activate Window (ns)	40 - 100	Default: 40
tPPD, Precharge to Precharge Delay (cycles)	4 - 7	Default: 4
tRAS, Row Active Time (ns)	42 - 100	Default: 42
tRCD, RAS-to-CAS Delay (ns)	18 - 100	Default: 18
tRPab, Row Precharge Time (All Banks) (ns)	21 - 100	Default: 21
tRPpb, Row Precharge Time (Single Bank) (ns)	18 - 100	Default: 18
tRRD, Active Bank-A to Active Bank-B (ns)	10 - 100	Default: 10
tRTP, Internal Read To Precharge Delay (ns)	7.5 - 100	Default: 7.5
tSR, Minimum Self Refresh Time (ns)	15 - 100	Default: 15
tWR, Write Recovery Time (ns)	18 - 60	Default: 18
tWTR, Write-To-Read Delay (ns)	10 - 60	Default: 10

Table 29: AXI 0 and AXI 1 Tabs

Parameter	Choices	Notes
Enable Target 0 Enable Target 1	On or off	Turn on to enable the AXI 0 interface. Turn on to enable the AXI 1 interface.
AXI Clock Input Pin Name	User defined	Specify the name of the AXI input clock pin.
Invert AXI Clock Input	On or off	Turn on to invert the AXI clock.
AXI Reset Pin Name	User defined	Specify the name of the AXI reset clock pin or use the default.
Read Address Channel tab Write Address Channel tab Write Response Channel tab Read Data Channel tab Write Data Channel tab	User defined	These tabs defines the AXI signal names for the channels. Efinix recommends that you use the default names.

Table 30: Controller Status Tab

Option	Choices	Notes
Invert Controller Status Clock Pin	On or off	Turn on if you want to invert the clock. Default: off
<description> Pin Name	User defined	Controller status pins. You can use the default names or specify your own.

Table 31: Pin Swizzling Tab

Option	Choices	Notes
Enable Package Pin Swapping	On or off	Turn on if you want to swap package pins. Default: off
DQ/DM Pin Swizzle Group*n*	–	Drag and drop pins in the DRAM column to swap DQ/DM pins. DQ Width = 16 : *n* is 0 and 1 DQ Width = 32 : *n* is 0, 1, 2 and 3
Address Pin Swizzle	–	Drag and drop pins in the DRAM column to swap address pins.

Design Check: DDR Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

ddr_rule_resource (error)

Message	Resource <name> is not a valid DDR device instance
To fix	The resource you specified does not exist. Check whether you have a typo in the resource name.

ddr_rule_invalid_pins (error)

Message	Invalid pin names found: <pin names>
To fix	The pin name you entered has illegal characters. Rename the pin.

ddr_rule_axi0_empty_pins (warning)

Message	Empty pin names found
To fix	You get this error when you add the DDR block but have not yet specified the **AXI Input Clock Pin Name** in the **AXI0** tab. If you do not want to use AXI0, turn off the **Enable Target 0** option in the **AXI0** tab.

ddr_rule_axi1_empty_pins (warning)

Message	Empty pin names found
To fix	You get this error when you add the DDR block but have not yet specified the **AXI Input Clock Pin Name** in the **AXI1** tab. If you do not want to use AXI0, turn off the **Enable Target 1** option in the **AXI1** tab.

ddr_rule_ctrl_reg_empty_pins (warning)

Message	Empty pin names found
To fix	You get this error when you add the DDR block but have not yet specified the **Controller Status Clock Pin Name** in the **Controller Status** tab.

ddr_rule_config_empty_pins (error)

Message	Empty pin names found
To fix	If you have names for any of the pins CFG_DONE, CFG_START, CFG_RESET, or CFG_SEL in the **Config Controller** tab, then you must have names for all of them. CFG_PHY_RSTN is optional.

ddr_rule_controller_clk (error)

Message	Clock to Controller pin name need to be specified with use of other Controller pins
To fix	You need to enter a name in the **Controller Status tab** > **Controller Status Clock Pin Name** field.

ddr_rule_reset_pin (error)

Message	Both Reset DDR PHY and Controller Reset pin names need to be specified if used
To fix	Enter both pin names.

ddr_rule_pll_feedback (error)

Message	Feedback mode for PLL <name> can only be set to local when DDR is configured
To fix	You cannot use **core** or **external** modes. Change the mode in the PLL Clock Calculator.

ddr_rule_phy_clock (error)

Message	Output clock 4 in PLL resource <res_name> for PHY Clock not configured
To fix	Use PLL_TL2 CLKOUT4 to drive the DDR controller.

ddr_rule_phy_clock (warning)

Message	PLL clock driving DDR with non-external clock source is not recommended
To fix	You get this warning if you do not connect the reference clock to an I/O pad. The PLL reference clock must be driven by I/O pads. (Using the clock tree may induce additional jitter and degrade the DDR performance.)

ddr_rule_ref_clock_freq (warning)

Message	PLL output frequency driving DDR is <#>MHz (max: <#>MHz)
To fix	Change the setting for the PLL output clock so that the frequency is lower than that maximum range.

ddr_rule_data_width (error)

Message	Selected Data Width <#> is not supported in device
To fix	The data width is dependent on the FPGA and package you choose. Refer to the data sheet to see which packages support the data width you want to use.

ddr_rule_clk_conn_type (error)

Message	Connection type of PLL <pll instance name> with output clock <pll outclk pin name> that is driving DDR cannot be set to rclk
To fix	Change the connection type of the PLL output clock to gclk.

ddr_rule_pll_output_clock (error)

Message	PLL Output clock <clk_num> phase shift should be 0 when DDR is configured
To fix	Set the PLL output clock <clk_num> phase shift to 0.

Message	Output clock 3 for PLL <pll_inst_name> should be used for configuring the high speed DDR clock
To fix	Enable the PLL's output clock 3.

Message	Frequency output clock 3 for PLL <pll_inst_name> should be 2 * Frequency of output clock 4
To fix	Change the PLL's output clock 3 and 4 frequencies so that output clock 3 frequency = 2 * output clock 4 frequency.

ddr_rule_pin_swap (error)

Message	Invalid pin name is set for DQ/DM Pin Swizzle (Group<#>: <list of invalid DQ/DM pin names>)
To fix	You get this error if you manually edit the **<project>.peri.xml** file wrongly. The DQ and DM pin names must be DQ<#> or DM<#> respectively.

Message	DQ/DM Pins has been used. (Group<#>: <list of invalid DQ/DM pin names>)
To fix	You get this error if you manually edit the **<project>.peri.xml** file wrongly. You cannot use more than one similar DQ/DM pin.

Message	Invalid pin name is set for Address Pin Swizzle. (FPGA: <list of invalid CA pin names>)
To fix	You get this error if you manually edit the **<project>.peri.xml** file wrongly. The CA pin names must be CA<#>.

Message	Address pin has been used. (FPGA: <list of invalid CA pin names>)
To fix	You get this error if you manually edit the **<project>.peri.xml** file wrongly. You cannot use more than one similar CA pin.

ddr_rule_pll_non_frac (error)

Message	PLL {pll_inst_name} driving DDR should disable fractional mode
To fix	Disable fractional mode for the PLL instance driving the DDR instance's clock source.

Message	Found {number of pins} DDR pin driven by PLL in fractional mode: {Pin names}
To fix	Disable fractional mode for the PLL instance driving those pins connected to DDR instance's AXI clock input.

Chapter 5

GPIO Interface

Contents:

Titanium FPGAs have general-purpose I/O (GPIO) pins that allow the FPGA to communicate with other components on your circuit board. When you create your RTL design in the Efinity® software, you use the Interface Designer to add GPIO blocks for each input, output, or bi-directional pin in your design.

Titanium GPIO pins have various features, depending on the position of the pin and which package you are using. Refer to the Resource Assigner in the Interface Designer for the features of the GPIO pin you want to use.

- GPIO that provide normal functionality
- GPIO with the double-data I/O (DDIO) feature that can capture twice the data
- HSIO as GPIO where the HSIO pin acts as a GPIO

The following sections describe the GPIO interface and how to use it in your design.

Types of GPIO

The Titanium FPGA supports two types of GPIO:

- *High-voltage I/O (HVIO)*—Simple I/O blocks that can support single-ended I/O standards.
- *High-speed I/O (HSIO)*—Complex I/O blocks that can support single-ended and differential I/O functionality.

The I/O logic comprises three register types:

- *Input*—Capture interface signals from the I/O before being transferred to the core logic
- *Output*—Register signals from the core logic before being transferred to the I/O buffers
- *Output enable*—Enable and disable the I/O buffers when I/O used as output

The HVIO supports the following I/O standards.

Table 32: HVIO Supported Standards

Standard	VCCIO33 (V)	When Configured As
LVTTL 3.3 V	3.3	GPIO
LVTTL 3.0 V	3.0	GPIO
LVCMOS 3.3 V	3.3	GPIO
LVCMOS 3.0 V	3.0	GPIO
LVCMOS 2.5 V	2.5	GPIO
LVCMOS 1.8 V	1.8	GPIO

Important: Efinix recommends that you limit the number of 3.0/3.3 V HVIO and 2.5 V HVIO as bidirectional or output to 6 per bank to avoid switching noise. The Efinity® software issues a warning if you exceed the recommended limit.

The HSIO supports the following I/O standards.

Table 33: HSIO Supported I/O Standards

Standard	VCCIO (V)		VCCAUX (V)	VREF (V)	When Configured As
	TX	RX			
LVCMOS 1.8 V	1.8	1.8	1.8	–	GPIO
LVCMOS 1.5 V	1.5	1.5	1.8	–	GPIO
LVCMOS 1.2 V	1.2	1.2	1.8	–	GPIO
HSTL/Differential HSTL 1.8 V SSTL/Differential SSTL 1.8 V	1.8	1.8	1.8	0.9	GPIO
HSTL/Differential HSTL 1.5 V SSTL/Differential SSTL 1.5 V	1.5	1.5, 1.8[9]	1.8	0.75	GPIO
SSTL/Differential SSTL 1.35 V	1.35	1.35, 1.5, 1.8[9]	1.8	0.675	GPIO
HSTL/Differential HSTL 1.2 V SSTL/Differential SSTL 1.2 V	1.2	1.2, 1.35, 1.5, 1.8[9]	1.8	0.6	GPIO
LVDS/RSDS/mini-LVDS	1.8	1.5, 1.8[9]	1.8	–	LVDS
Sub-LVDS	1.8	1.5, 1.8[9]	1.8	–	Sub-LVDS
MIPI	1.2	1.2	1.8	–	MIPI Lane
SLVS	1.2	1.2	1.8	–	SLVS

The differential receivers are powered by VCCAUX, which gives you the flexibility to choose the VCCIO you want to use. However, you must comply to the requirements stated in the previous table.

[9] To prevent pin leakage, you must ensure that the voltage at the pin does not exceed VCCIO.

Features for HVIO and HSIO Configured as GPIO

The following table describes the features for HVIO and HSIO configured as GPIO.

Table 34: Features for HVIO and HSIO Configured as GPIO

Feature	HVIO	HSIO Configured as GPIO
Double-data I/O (DDIO)	✓	✓
Dynamic pull-up	–	✓
Pull-up/Pull-down	✓	✓
Slew-Rate Control	–	✓
Variable Drive Strength	✓	✓
Schmitt Trigger	✓	✓
1:4 Serializer/Deserializer (Full rate mode only)	–	✓
Programmable Bus Hold	–	✓
Static Programmable Delay Chains	✓	✓
Dynamic Programmable Delay Chains	–	✓

Table 35: GPIO Modes

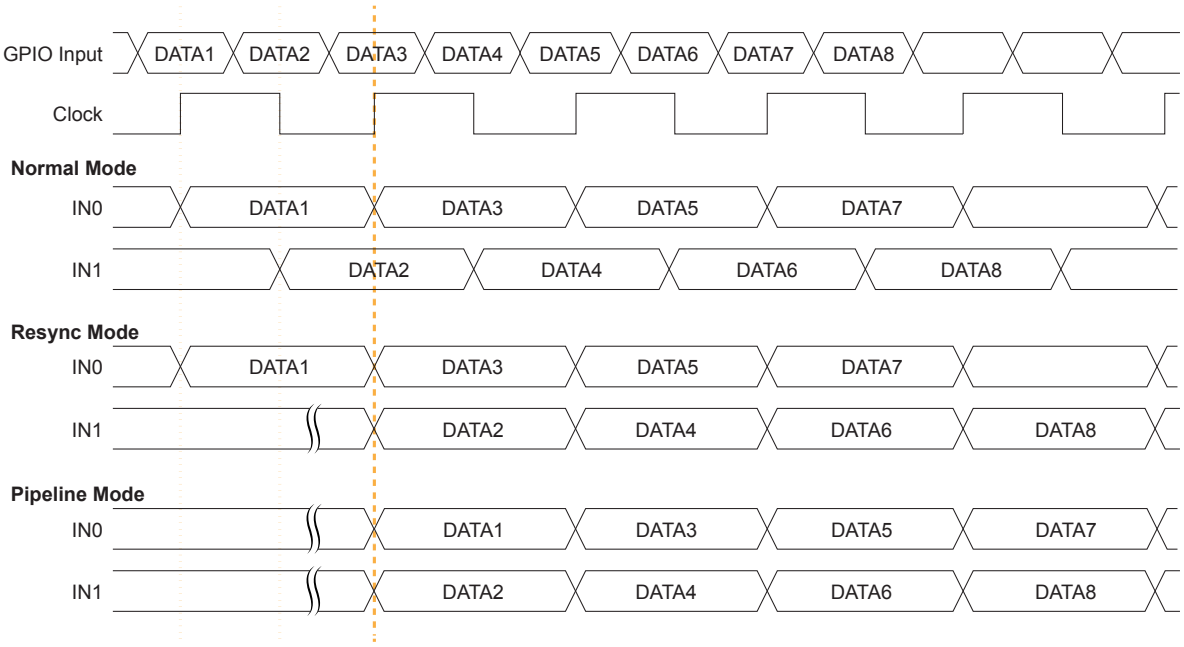
GPIO Mode	Description
Input	Only the input path is enabled; optionally registered. If registered, the input path uses the input clock to control the registers (positively or negatively triggered). Select the alternate input path to drive the alternate function of the GPIO. The alternate path cannot be registered. In DDIO mode, two registers sample the data on the positive and negative edges of the input clock, creating two data streams.
Output	Only the output path is enabled; optionally registered. If registered, the output path uses the output clock to control the registers (positively or negatively triggered). The output register can be inverted. In DDIO mode, two registers capture the data on the positive and negative edges of the output clock, multiplexing them into one data stream.
Bidirectional	The input, output, and OE paths are enabled; optionally registered. If registered, the input clock controls the input register, the output clock controls the output and OE registers. All registers can be positively or negatively triggered. Additionally, the input and output paths can be registered independently. The output register can be inverted.
Clock output	Clock output path is enabled.

Double-Data I/O

Titanium FPGAs support double data I/O (DDIO) on input and output registers. In this mode, the DDIO register captures data on both positive and negative clock edges. The core receives 2 bit wide data from the interface.

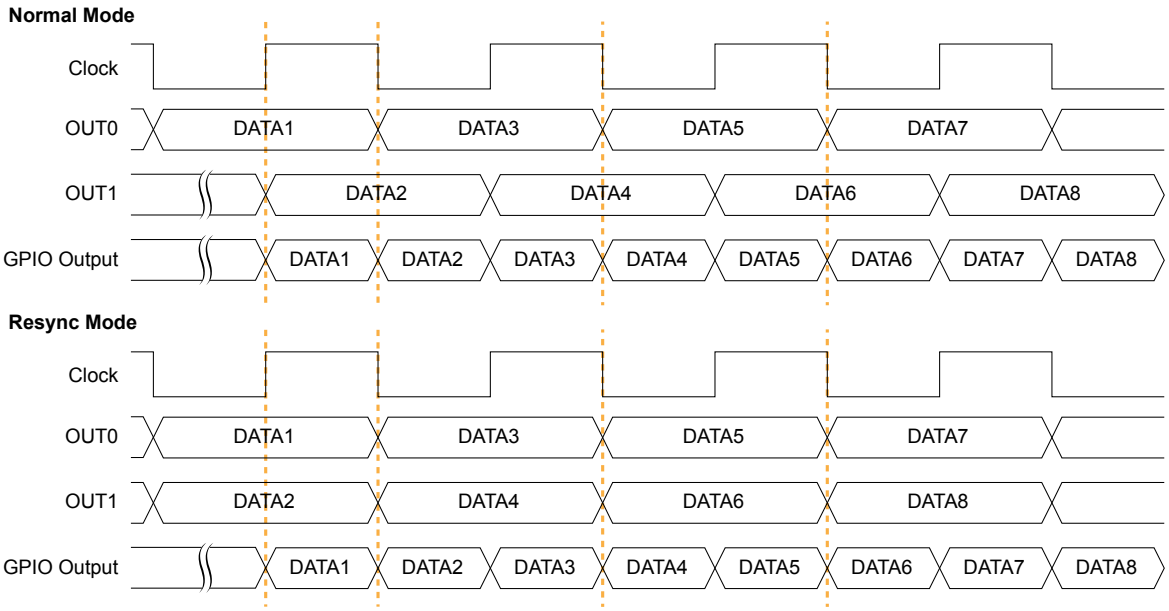
In normal mode, the interface receives or sends data directly to or from the core on the positive and negative clock edges. In resync and pipeline mode, the interface resynchronizes the data to pass both signals on the positive clock edge only.

Figure 19: DDIO Input Timing Waveform



In resync mode, the IN1 data captured on the falling clock edge is delayed one half clock cycle.
In the Interface Designer, IN0 is the HI pin name and IN1 is the LO pin name.

Figure 20: DDIO Output Timing Waveform



In the Interface Designer, OUT0 is the HI pin name and OUT1 is the LO pin name.

Programmable Delay Chains

The HVIO and HSIO configured as GPIO support programmable delay chain. In some cases you can use static and dynamic delays at the same time.

Table 36: Programmable Delay Support

GPIO Type	Delay Steps	
	Static Delay	Dynamic Delay
Single-Ended		
HVIO input	16	N/A
HVIO output	16	N/A
HSIO P pin input	16	64
HSIO P pin output	16	N/A
HSIO N pin input	16	N/A
HSIO N pin output	16	N/A
Differential		
HSIO TX	64	N/A
HSIO RX	64[10]	64[10]

Learn more: Refer to the device data sheet for the delay step size.

[10] You cannot use the static delay and dynamic delay simultaneously.

About the HVIO Interface

The HVIOs are grouped into banks. Each bank has its own VCCIO33 that sets the bank voltage for the I/O standard. Each HVIO consists of I/O logic and an I/O buffer. I/O logic connects the core logic to the I/O buffers. I/O buffers are located at the periphery of the device.

Figure 21: HVIO Interface Block

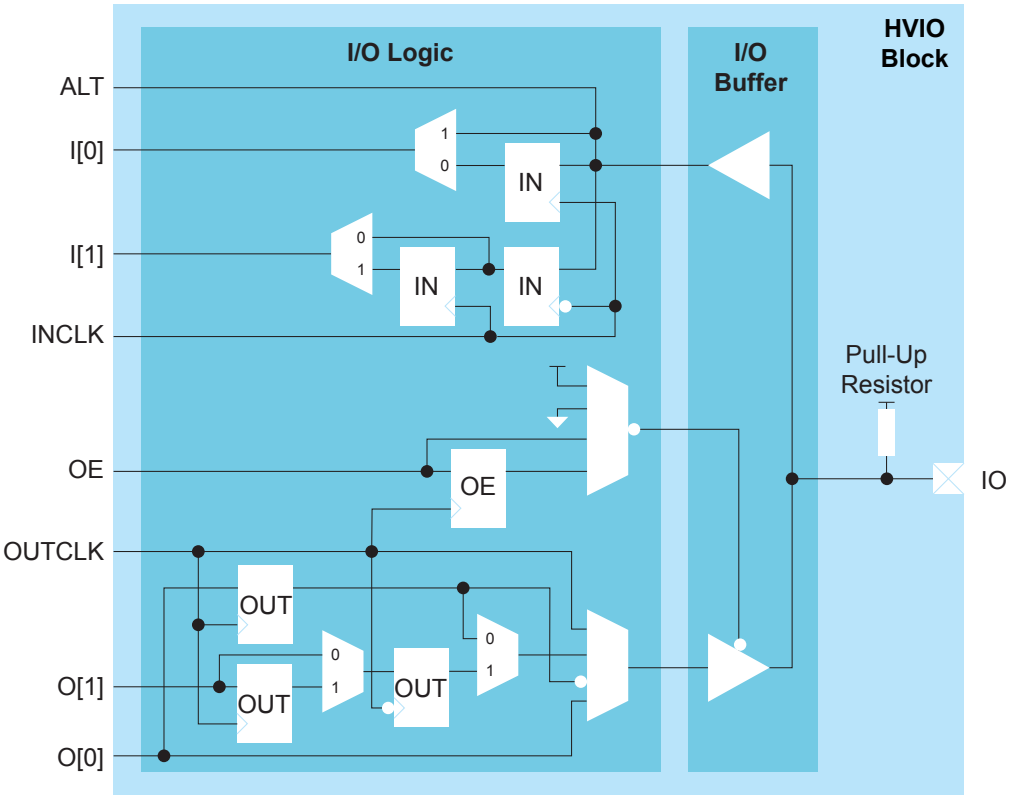


Table 37: HVIO Signals (Interface to FPGA Fabric)

Signal	Direction	Description
I[1:0]	Output	Input data from the HVIO pad to the core fabric. I[0] is the normal input to the core. In DDIO mode, I[0] is the data captured on the positive clock edge (HI pin name in the Interface Designer) and I[1] is the data captured on the negative clock edge (LO pin name in the Interface Designer).
ALT	Output	Alternative input connection (in the Interface Designer, **Register Option** is **none**). HVIO only support pll_clkin as the alternative connection.
O[1:0]	Input	Output data to HVIO pad from the core fabric. O[0] is the normal output from the core. In DDIO mode, O[0] is the data captured on the positive clock edge (HI pin name in the Interface Designer) and O[1] is the data captured on the negative clock edge (LO pin name in the Interface Designer).
OE	Input	Output enable from core fabric to the I/O block. Can be registered.
OUTCLK	Input	Core clock that controls the output and OE registers. This clock is not visible in the user netlist.
INCLK	Input	Core clock that controls the input registers. This clock is not visible in the user netlist.

Table 38: HVIO Pads

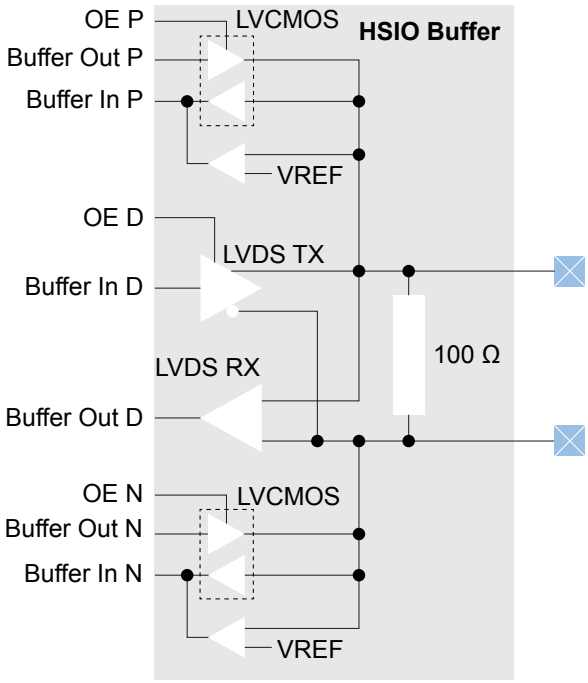
Signal	Direction	Description
IO	Bidirectional	HVIO pad.

About the HSIO Interface

Each HSIO block uses a pair of I/O pins as one of the following:

- *Single-ended HSIO*—Two single-ended I/O pins (LVCMOS, SSTL, HSTL)
- *Differential HSIO*—One differential I/O pins:
  - Differential SSTL and HSTL
  - LVDS—Receiver (RX), transmitter (TX), or bidirectional (RX/TX)
  - MIPI lane I/O—Receiver (RX) or transmitter (TX)

Figure 22: HSIO Buffer Block Diagram



Important: When you are using an HSIO pin as a GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and LVDS or MIPI lane pins. This rule applies for pins on each side of the device (top, bottom, left, right). This separation reduces noise. The Efinity software issues an error if you do not leave this separation.

HSIO Configured as GPIO

You can configure each HSIO block as two GPIO (single-ended) or one GPIO (differential).

Figure 23: I/O Interface Block

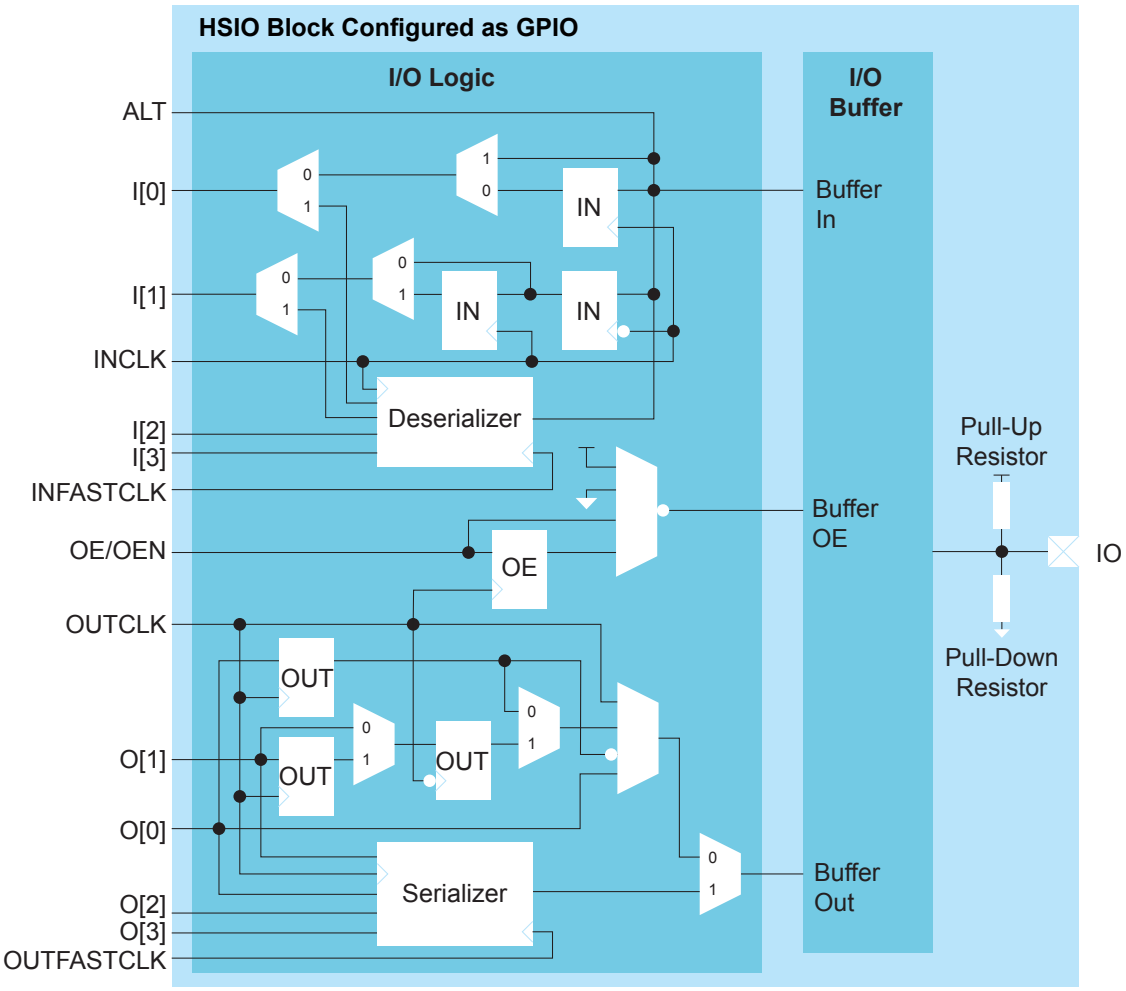


Table 39: HSIO Block Configured as GPIO Signals (Interface to FPGA Fabric)

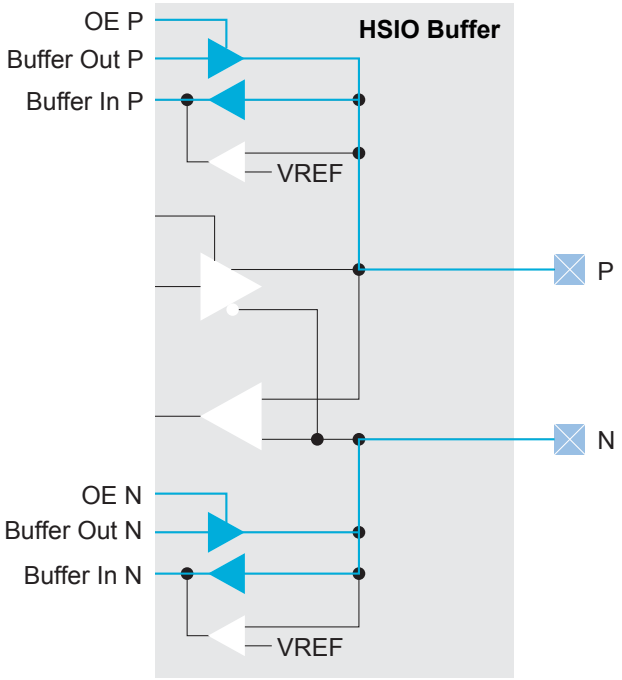
Signal	Direction	Description
I[3:0]	Output	Input data from the pad to the core fabric. I[0] is the normal input to the core. In DDIO mode, I[0] is the data captured on the positive clock edge (HI pin name in the Interface Designer) and I[1] is the data captured on the negative clock edge (LO pin name in the Interface Designer). When using the deserializer, the first bit is on I[0] and the last bit is on I[3].
ALT	Output	Alternative input connection for GCLK, PLL_CLKIN, RCLK, PLL_EXTFB, and VREF. (In the Interface Designer, **Register Option** is **none**).
O[3:0]	Input	Output data to GPIO pad from the core fabric. O[0] is the normal output from the core. In DDIO mode, O[0] is the data output on the positive clock edge (HI pin name in the Interface Designer) and O[1] is the data output on the negative clock edge (LO pin name in the Interface Designer). When using the serializer, the first bit is on O[0] and the last bit is on O[3].
OE/OEN	Input	Output enable from core fabric to the I/O block. Can be registered. OEN is used in differential mode. Drive it with the same signal as OE.
DLYCLK	Input	Delay clock for dynamic delay, sampled on the negative edge. In serializer mode, this clock must be the same clock as INCLK.
DLY_ENA	Input	(Optional) Enable the dynamic delay control.
DLY_INC	Input	(Optional) Dynamic delay control. When DLY_ENA = 1, 1: Increments 0: Decrements The updated delay count takes effect approximately 5 ns after the rising edge of the clock.
DLY_RST	Input	(Optional) Reset the delay counter.
OUTCLK	Input	Core clock that controls the output and OE registers. This clock is not visible in the user netlist.
OUTFASTCLK	Input	Core clock that controls the output serializer.
INCLK	Input	Core clock that controls the input registers. This clock is not visible in the user netlist.
INFASTCLK	Input	Core clock that controls the input serializer.

Table 40: GPIO Pads

Signal	Direction	Description
IO (P and N)	Bidirectional	GPIO pad.

The signal path from the pad through the I/O buffer changes depending on the I/O standard you are using. The following figures show the paths for the supported standards. The blue highlight indicates the path.

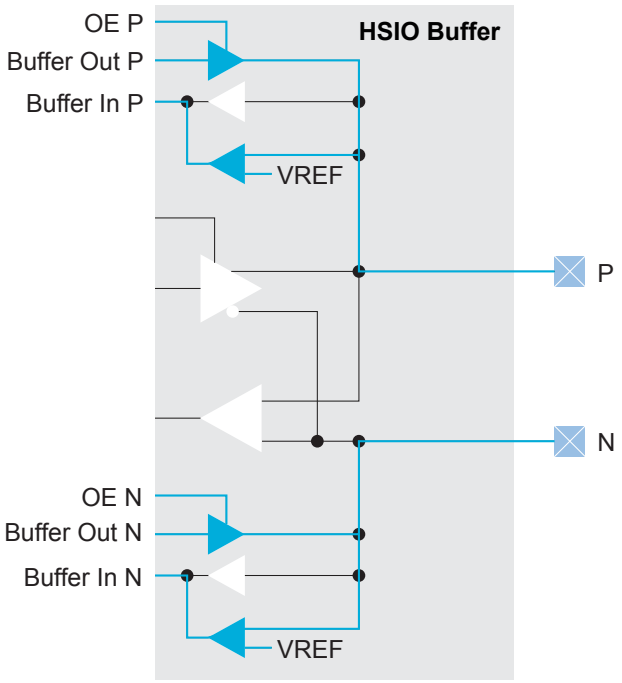
Figure 24: I/O Buffer Path for LVCMOS



When using an HSIO with the HSTL or SSTL I/O standards, you must configure an I/O pad of the standard's input path as a `VREF` pin. There is one programmable `VREF` per I/O bank.

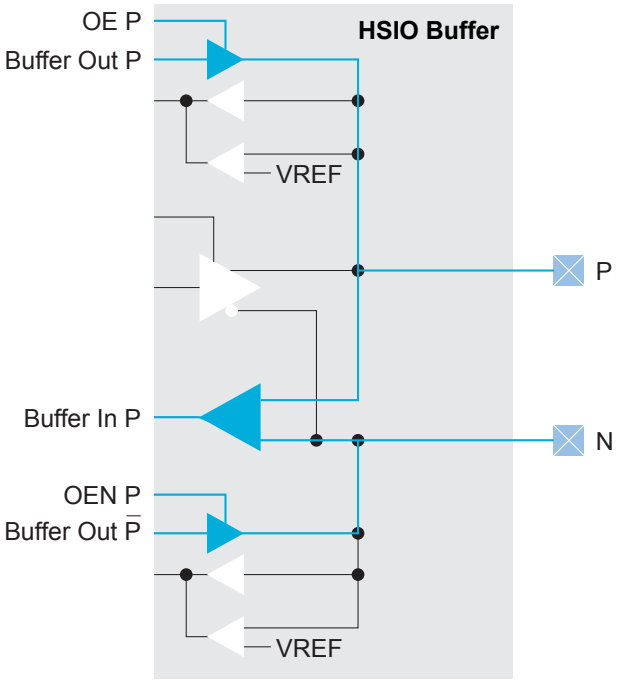
Important: When configuring an I/O pad of the standard's input path as a `VREF` pin, you must use the `VREF` from the same physical I/O bank even when the I/O banks are merged to share a common VCCIO pin.

Figure 25: I/O Buffer Path for HSTL and SSTL



When using an HSIO with the differential HSTL or differential SSTL standard, you must use both GPIO resources in the HSIO. You use the core interface pins associated with the P resource.

Figure 26: I/O Buffer Path for Differential HSTL and SSTL



Using the GPIO Block

This block defines the functionality of the general-purpose I/O (GPIO) pins. The mode you select determines the GPIO capabilities and which settings you can configure. GPIO modes are: input, output, inout, clkout, and none.

You can assign GPIO to HVIO or HSIO resources. These resources support different I/O standards and have different features. When you check the interface design, the software compares your selections to the resource you assigned to the GPIO block. If the resource does not support your selection(s), the software reports it.

Create a GPIO

To create a new GPIO block, select GPIO in the Design Explorer and then click the Create Block button.

1. Specify the instance name.
2. Choose the Mode (input, output, inout, clkout, or none).
3. Set the options as described in the following sections.
4. Assign a resource for the signal using the Resource Assigner.

Note: You can set the default state of unused GPIO. Click the **GPIO(*n*)** category under Design Explorer. In the Block Editor to the right, select the unused state (**input with weak pull up** or **input with weak pull down**).

Note: When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO pins in the same bank. This separation reduces noise. The Efinity software issues an error if you do not leave this separation.

Input Mode

Use **input** mode for input signals.

Table 41: Input Mode Options

Option	Choices	Description
Connection Type	normal, gclk, pll_clkin, pll_extfb, mipi_clkin, pcie_perstn,[11] rclk, vref	Some pins have alternate functions, and you use this option to choose the function. (This option only applies to pins that have alternate functions. Refer to the data sheet for your FPGA for pin information.) For example, a PLL can use a GPIO with an alternate connection type as a reference clock. **Note:** If you set the connection type to **pll_clkin** or **mipi_clkin**, the signal is also available as a regular input to the core.
Register Option	register, none	Choose whether the input is registered. If you choose **register**: • Define an input clock pin name. • Turn clock inversion on or off. • Under **Double Data I/O Option**, select one of the following: **none** — Do not use double data I/O. **normal** — Data is passed to the core on both the positive and negative clock edges **resync** — Data is resynchronized to pass both data signals on the positive clock edge. `<pin name>_hi` is the positive edge and `<pin name>_lo` is the negative edge. **pipeline** — Similar to **resync** except the data for both signals starts on the same clock edge. • To use the deserializer, turn on **Enable Deserialization** and specify the serial clock pin name. (You cannot use the deserializer with DDIO.)
Pull Option	none, weak pullup, weak pulldown, dynamic	Specify if you want a pull option. If you choose **dynamic**, you must also specify the **Dynamic Pull Up Enable Pin Name**.
Enable Schmitt Trigger	On or off	Optionally enable a Schmitt trigger.
Enable Bus Hold	On or off	Optionally enable a bus hold.
Static Delay Settting	Integer 0-15	For single-ended only. Choose the amount of static delay, each step adds approximately 60 ps of delay. You can only set the static delay for individual signals, not buses.
	0 - 63	For differential only: 64 steps with approximately 25 ps of delay per step. You cannot use the static delay and dynamic delay simultaneously.

[11] Only available for FPGAs with transceivers.

Option	Choices	Description
Enable Dynamic Delay	On or off	For inputs, 64 steps with approximately 25 ps of delay per step. If you enable this option, specify the enable, reset, and control pins as well as the clock pin. You can ony set the dynamic delay for individual signals, not buses. A clock is required when using the dynamic delay. If the input register is used, the clock is the same as the input register. Otherwise, you have to define a clock in the dynamic delay group box. The delay is updated on the rising clock edge of DLYCLK.

Output Mode

Use **output** mode for output signals.

Table 42: Output Mode Options

Option	Choices	Description
Constant Output	none, 1, 0	Choose whether the output is VCC (**1**) or GND (**0**). Otherwise, leave this option as none.
Register Option	none, register, inv_register	Choose whether the output is registered or has an inverted register. If you choose **register**: • Define an output clock pin name. • Turn clock inversion on or off. • Under **Double Data I/O Option**, select one of the following: **none** — Do not use double data I/O. **normal** — Data is passed to the core on both the positive and negative clock edges **resync** — Data is resynchronized to pass both data signals on the positive clock edge. `<pin name>_hi` is the positive edge and `<pin name>_lo` is the negative edge. • To use the serializer, turn on **Enable Serialization** and specify the serial clock pin name. (You cannot use the serializer with DDIO.) The invert register option (**inv_register**) does not support DDIO.
Drive Strength	Depends on I/O standard	Choose the drive strength current in mA. The HVIO and HSIO have different drive strength options depending on the I/O standard you choose. If you change the I/O standard, the Interface Designer resets the drive strength setting if the value is out of range for the selected standard.
Enable Fast Slew Rate	On or off	Optionally enable slew rate.
Static Delay Settting	0 - 63	Choose the amount of static delay, each step adds approximately 60 ps of delay. You can only set the static delay for individual signals, not buses.

Inout Mode

Use **inout** mode for bidirectional signals. Inout mode has the same options for the input and output as the input and output modes.

Inout mode also has an output enable signal (optionally registered) to enable or disable the output buffer. The pin name you specify should be the same as the one you use in your RTL

design. Setting the output enable signal to high (“1”) in your RTL design enables the output buffer.

Learn more: For information on how to create a tri-state buffer, refer to “How do I create a Tri-State Buffer" in the Support Center Knowledgebase.

Clock Output Mode

Use **clkout** mode for clock output signals. You do not need to name the pin, but you do need to specify the output clock **Pin Name**.

None

Use **none** for unused signals. Specify whether the unused signal should have a weak pullup (default) or pulldown.

Using the GPIO Bus Block

The GPIO bus block is an easy way to add a group of GPIO blocks and make settings for the signal group.

1. Click **Create New Bus**. The Add New Bus wizard opens.
2. Specify a bus name, the width, and the mode (input, output, or inout) and click **Next**.
3. The wizard displays options for input, output, or inout, depending on the mode you selected. Refer to Using the GPIO Block on page 71 for a description of these options. Make your selections and click **Next**.
4. Review the bus properties and click **Finish**. The software adds the new bus under GPIO.

After you create a bus, you can make additional settings for each signal.

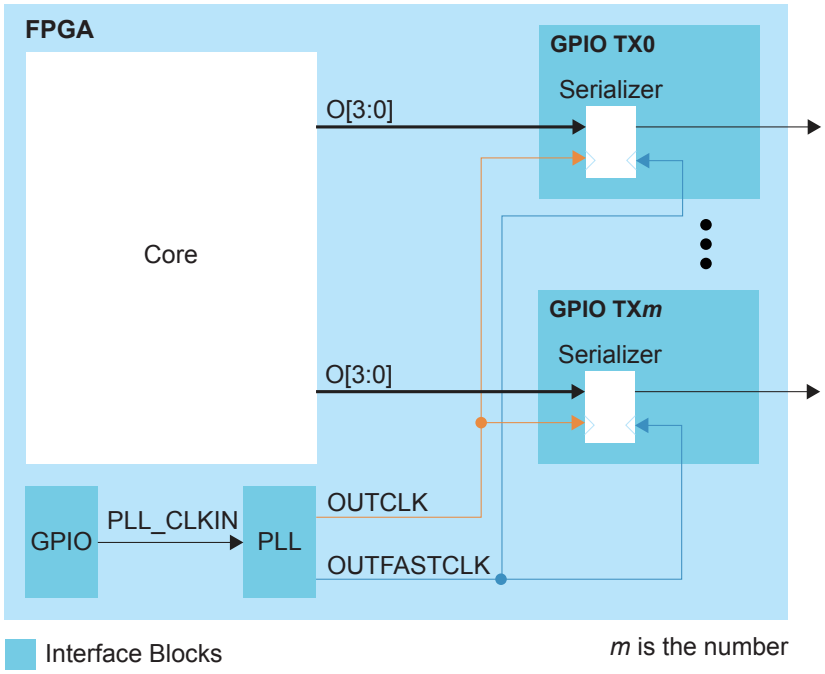
1. Expand **GPIO > <bus name>**.
2. Make any block-specific settings in the Block Editor.
3. Assign a resource for the signal using the Resource Assigner.
4. Save.

Note: Any changes that you make to individual bus members are over written if you later edit the bus.

Create a TX Serializer Interface

The following figure shows a completed TX serializer interface, the serialization width is always 4 and *m* is the number of TX lanes.

Figure 27: Complete TX Serializer Interface Block Diagram



Follow these steps to build this interface using the Efinity® Interface Designer.

1. Add a PLL block with the following settings:

Option	Description
Resource	You can use any PLL resource.
Reference Clock Mode	Any
Reference Clock Frequency	Any
Output Clock	Define the output clocks so that you have one for the fast clock (serial) and one for the slow clock (parallel). The fast clock (OUTFASTCLK) should be 4 times faster than the slow clock (OUTCLK). The serial clock phase shift should be between 45 and 135 degrees.

2. Add a GPIO block with these settings to provide the reference clock input to the PLL:

Option	Description
Mode	Input
Pin Name	Any
Connection Type	pll_clkin
GPIO Resource	Assign the dedicated PLL_CLKIN pin that corresponds to the PLL you chose.

3. Add a GPIO block with these settings:

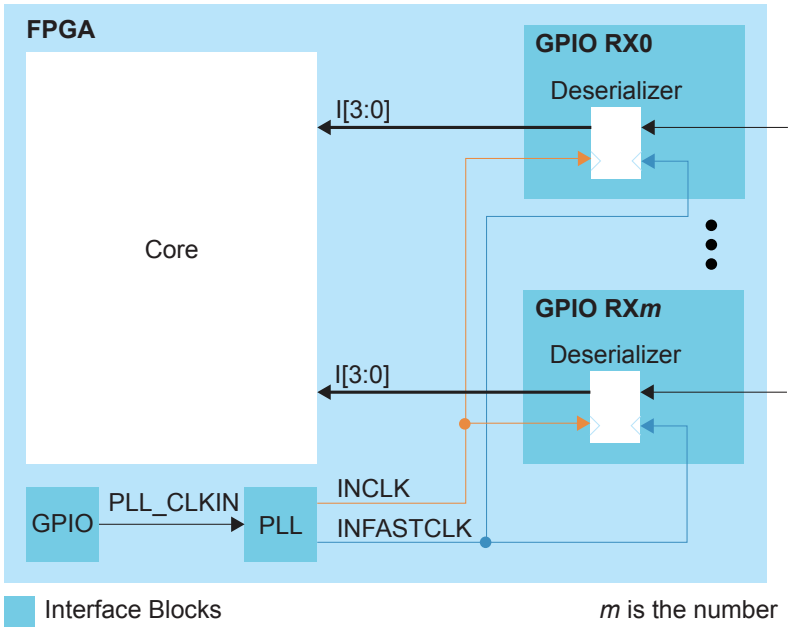
Option	Description
Mode	output
Register Option	register
Enable Serialization	Turn on
Clock Pin Name	Use the slow clock output name that corresponds to the PLL you chose.
Serial Clock Pin Name	Use the fast clock output name that corresponds to the PLL you chose.

4. Repeat step 3 for each TX serializer you want to implement.

Create a RX Deserializer Interface

The following figure shows a completed RX deserializer interface, the deserialization width is 4 and *m* is the number of RX lanes.

Figure 28: Complete RX ISerializer nterface Block Diagram



Follow these steps to build this interface using the Efinity® Interface Designer.

1. Add a PLL block with the following settings:

Option	Description
Resource	You can use any PLL resource.
Reference Clock Mode	Any
Reference Clock Frequency	Any
Output Clock	Define the output clocks so that you have one for the fast clock (serial) and one for the slow clock (parallel). The fast clock (INFASTCLK) should be 4 times faster than the slow clock (INCLK). . The serial clock phase shift should be between 45 and 135 degrees.

2. Add a GPIO block with these settings to provide the reference clock input to the PLL:

Option	Description
Mode	Input
Pin Name	Any
Connection Type	pll_clkin
GPIO Resource	Assign the dedicated PLL_CLKIN pin that corresponds to the PLL you chose.

3. Add a GPIO block with these settings:

Option	Description
Mode	input
Register Option	register
Enable Serialization	Turn on
Clock Pin Name	Use the slow clock output name that corresponds to the PLL you chose.
Serial Clock Pin Name	Use the fast clock output name that corresponds to the PLL you chose.

4. Repeat step 3 for each RX deserializer you want to implement.

Design Check: GPIO Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

bus_rule_members_consistent (error)

Message	Bus <name> has mismatch properties with its members Members of bus <name> has inconsistent shared pin/register settings with <member>: <inconsistent_members_name>
To fix	The properties you have set for the bus are inconsistent with the settings for the bus members. Review the settings and fix any mismatches.

bus_rule_name (error)

Message	Bus name is empty Valid characters are alphanumeric characters with dash and underscore only
To fix	Specify a valid bus name.

gpio_rule_inst_name (error)

Message	Instance name is empty
To fix	Specify a valid bus name.

Message	GPIO name <instance name> is used
To fix	The GPIO name <instance name> is already in use. Specify a different instance name.

gpio_rule_input_mode (error)

Message	For input mode, input must be configured
To fix	You need to configure the input parameters.

Message	For input mode, input pin name must be configured
To fix	Specify a name for the input pin.

Message	Input pin name with square bracket does not match a bus name with index
To fix	You probably used a bracket [or] in the pin name. Rename the pin without brackets.

gpio_rule_input_register (error)

Message	Input clock pin name is empty Input clock pin name has illegal character
To fix	If you are using the register option, you need to specify a valid clock pin name.

gpio_rule_input_register (warning)

Message	Input clock pin name is empty Input clock pin name has illegal character
To fix	If you have a GPIO block in inout mode, you get this message if you do not specify an input pin name or if you use invalid characgters in the name. You should specify a name to use the pin as bidirectional.

Message	Alternate input connection cannot be registered
To fix	When you are using an alternate connection type. you cannot use the register. Set **Register Option** to **none**.

gpio_rule_output_mode (error)

Message	For output mode, output must be configured
To fix	You need to configure the output parameters.

Message	For output mode without constant output, output pin name must be configured
To fix	For GPIO output blocks, you can drive them as 0 or a 1 with the **Constant Output** option. In that mode, you do not need to specify an output pin name. If you are using **Constant Output** set to **none** (as you would for a regular output pin), you need to specify a pin name.

Message	Output pin name with square bracket does not match a bus name with index
To fix	You probably used a bracket [or] in the pin name. Rename the pin without brackets.

gpio_rule_inout_mode (error)

Message	For inout mode, both output and output enable must be configured
To fix	When using a GPIO block in inout mode, you need to set the options for the output and output enable (as well as the input).

Message	For inout mode, both output pin name and output enable pin name must be configured
To fix	Specify a valid name for the output pin and output enable pin.

Message	Output enable pin name with square bracket does not match a bus name with index
To fix	You probably used a bracket [or] in the pin name. Rename the pin without brackets.

gpio_rule_clkout_mode (error)

Message	For clkout mode, output must be configured
To fix	When using a GPIO block in clkout mode, you need to set all of the options.

Message	For clkout mode, output clock pin name must be configured
To fix	When using a GPIO block in clkout mode, you need to specify a valid pin name for the ouput clock.

gpio_rule_output_clock (error)

Message	Output is registered but clock pin name is empty
To fix	If you choose **register** or **inv_register** as the **Register Option**, then you also need to specify an output clock pin name.

Message	Output enable is registered but clock pin name is empty
To fix	If you are using a GPIO in inout mode, you need to specify the output enable pin name if you set the output enable to **register**.

Message	Output is registered but clock pin name has illegal character Output enable is registered but clock pin name has illegal character
To fix	Use valid names for these pins.

Message	Output clock pin name is not the same as output enable clock pin name
To fix	For a GPIO block in inout mode, you need to use the same pin names for the output clock pin and the output enable clock pin.

Message	Output clock inversion is not the same as output enable clock inversion
To fix	For a GPIO block in inout mode, if you invert the output clock pin you also need to invert the output enable clock pin. Similarly, if you do not invert the output clock pin, you cannot invert the output enable clock pin.

gpio_rule_unused_mode (error)

Message	For unused (none) mode, its state needs to be configured
To fix	If you set a GPIO block mode to **none**, you need to set the **Unused State**. The default is input with weak pull-up. You can change this setting globally by clicking the GPIO category in the Design Explorer and setting the **Unused State** option.

gpio_rule_input_alt_conn (error)

Message	Connection type <type> is not supported by <resource>
To fix	If you want to use the alternate funciton of a GPIO block, you need to choose a resource that supports it. For example, global clock (GCLK) is only supported on the P pin. You can filter for resources by alternate function in the Resource Assigner.

Message	<resource> only supports normal connection type
To fix	You need to choose a different connection type or assign a different resource that supports the connection type you want to use. You can filter for resources by alternate function in the Resource Assigner.

gpio_rule_input_alt_conn (warning)

Message	Connection type <type> is not supported by <resource>
To fix	For a GPIO block in inout mode, you get a warning if you do not specify the input pin name. Specify the input pin name to use the block as bidirectional or leave it empty to use it as open-drain.

Message	<resource> only supports normal connection type
To fix	The software thinks you are creating an open-drain if you leave the input pin name empty, so this choice is valid, but it lets you know that you made this selection in case you really want to use it as bidirectional.

gpio_rule_ddio_resource (error)

Message	Double Data I/O must be assigned to resource that supports DDIO
To fix	To use the DDIO feature, you need to pick a resource that supports it. You can filter for resources by DDIO in the Resource Assigner **Features** column.

gpio_rule_ddio_resource (warning)

Message	Double Data I/O must be assigned to resource that supports DDIO
To fix	You get this error if you use a resource that does not support DDIO but have not yet specified an input pin name. Either turn off DDIO or choose a resource that supports it. You can filter for resources by DDIO in the Resource Assigner **Features** column.

gpio_rule_ddio_input (error)

Message	Input with DDIO requires register option to be set
To fix	If you are using DDIO, you must set the the **Register Option** to **register**.

gpio_rule_ddio_input (warning)

Message	Input with DDIO requires register option to be set
To fix	For a GPIO block in inout mode, if the **Double-Data I/O Option** is other than **none**, you need to choose **register** as the **Register Option**. You get this warning when you have set some options for the input pin but have not yet specified a pin name.

gpio_rule_ddio_output (error)

Message	Output with DDIO requires register option to be set
To fix	To use DDIO you must set the **Double-Data I/O Option** set to something other than **none**.

gpio_rule_ddio_pin_name (error)

Message	Double Data I/O must have both HI and LO input pin names defined
To fix	When using DDIO, you need to specify pin names for the **Pin Name (HI)** and **Pin Name (LO)**.

gpio_rule_ddio_pin_name (warning)

Message	Double Data I/O must have both HI and LO input pin names defined
To fix	For a GPIO block in inout mode, if the **Double-Data I/O Option** is other than **none**, you need to specify pin names for the **Pin Name (HI)** and **Pin Name (LO)**. You get this warning when you have set some options for the input pin but have not yet specified these pin names.

gpio_rule_alt_conn (warning)

Message	Connection type <type> must be used by valid PLL
To fix	The GPIO is connected to a PLL clock input but is the resource you assigned does not support the pll_clkin alternate function. Choose a different resource that supports it. You can filter resources by alternate function in the Resource Assigner.

Message	Connection type <type> cannot be used on an unbonded resource
To fix	You get this error if the resource you choose is not available in the FPGA/package combination you are using. Choose another resource.

Message	pll_clkin connection to PLL clock source not being used in <instance>
To fix	The GPIO block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. Make sure that the clock you are choosing in the PLL is associated with this GPIO's resource.

Message	pll_clkin connection to PLL clock source but none of the external clock source in PLL <instance> is selected
To fix	The GPIO block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. In the PLL block, choose external or dynamic as the Clock Source and make sure that the clock you are choosing is associated with this GPIO's resource.

Message	pll_clkin connection to PLL clock source but PLL Clock source on <instance> is set to core
To fix	The GPIO block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. In the PLL block, choose external or dynamic as the Clock Source and make sure that the clock you are choosing is associated with this GPIO's resource.

Message	pll_extfb connection to PLL external feedback pin on <instance> is not set to external
To fix	The GPIO block is set to be external feedback for the PLL (pll_extfb connection type) but the PLL is not configured to use it. In the PLL Clock Calculator, choose External as the Feedback Mode.

gpio_rule_sample_device (error)

Message	Unsupported features in ES device: <features>
To fix	You get this error when you try to use a feature that is not supported in an engineering sample (ES) version of the FPGA.

gpio_rule_resource (error)

Message	Resource name is empty Resource is not a valid GPIO device instance
To fix	You need to choose a valid resource.

gpio_rule_io_standard_bank (warning)

Message	Mismatch voltage in I/O standard assignment in bank (<voltage>) and instance (<io_std>)
To fix	You get this error when the voltage for the I/O bank does not match the I/O standard you chose for the GPIO. Either change the I/O bank voltage or choose a compatible I/O standard.

gpio_rule_io_standard_compatibility (error)

Message	I/O standard <value> is not supported in bank <bank>
To fix	You get this error when you choose an I/O standard that is not supported in the bank. You need to choose another I/O standard or pick a resource in another bank. See **Titanium I/O Banks** on page 30 for the voltages supported in each bank and **Types of GPIO** on page 60 for the I/O standards the GPIO support.

gpio_rule_drive_strength (error)

Message	Valid drive strength for <iostd> is: <list>
To fix	Choose the drive strength based on the recommendation in the message.

Message	Invalid drive strength <value> for <iostd>. Check for valid I/O standard
To fix	Confirm the drive strengths that are allowed for the I/O standard you want to use and then change the setting accordingly.

gpio_rule_ddio_serial (error)

Message	DDIO has to be none when serialization is enabled on both input and output DDIO on input has to be none when deserialization is enabled DDIO on output has to be none when serialization is enabled
To fix	You cannot use DDIO and serialization or deserialization at the same time. Turn one of them off.

gpio_rule_transmit_toggling (warning)

Message	Bank <name> has <int> GPIO in inout/output mode that exceed max limit of <int> which can result in LVTTL simultaneous switching noise
To fix	If you use more than 6 HVIO pins as GPIO in output or inout mode in the same bank, it can cause switching noise. Instead, use resources from another bank.

gpio_rule_serial_input_clk (error)

Message	Input clock inversion is not allowed with deserialization enabled
To fix	If you use deserialization, you cannot also invert the clock. Turn off the **Inverted** option.

gpio_rule_serial_output_clk (error)

Message	Output clock inversion is not allowed with serialization enabled
To fix	If you use serialization, you cannot also invert the clock. Turn off the **Inverted** option.

gpio_rule_static_input_delay (error)

Message	Static delay, <int> is outside of limit (0-15) for non-Differential HSTL/SSTL I/O Standard Static delay, <int> is outside of limit (0-63) for Differential HSTL/SSTL I/O Standard
To fix	The static input delay you selected is not valid. Use a number in the range specified in the message.

gpio_rule_io_standard_valid (error)

Message	I/O standard <value> is not supported in bank <bank>
To fix	You get this error when you choose an I/O standard that is not supported in the bank. You need to choose another I/O standard or pick a resource in another bank. See **Titanium I/O Banks** on page 30 for the voltages supported in each bank and **Types of GPIO** on page 60 fo rthe I/O standards the GPIO support.

gpio_rule_hsio_usage (error)

Message	HSIO resource <name> was assigned to GPIO, LVDS and MIPI LANE HSIO resource <name> was assigned to both GPIO and LVDS HSIO resource <name> was assigned to both GPIO and MIPI LANE
To fix	You get this error if you try to use the same resource for more than one block type. Remove blocks so that you only are only using the resource once.

gpio_rule_io_standard_stl (error)

Message	This resource is reserved as vref for bank <name>. Use a different resource to configure single ended HSTL/SSTL
To fix	Some resources can be used as the VREF for an I/O standard. If you are using an I/O standard that uses a VREF pin, you must use this resource as a VREF. Choose another resource for the GPIO function.

Message	GPIO <name> has to be configured as vref input mode to support I/O standard <iostd> GPIO <name> has to be configured as vref input connection type to support I/O standard <iostd> GPIO <name> has to be configured as vref input to support I/O standard <iostd>
To fix	If you are using an I/O standard that uses a VREF pin, you must use this resource as a VREF. Configure the GPIO as an input and choose **vref** as the **Connection Type**.

Message	I/O Standard <iostd> cannot be used due to unbonded vref resource on the same bank <name> I/O Standard <iostd> cannot be used due to vref resource not bonded out
To fix	If a VREF pin is not available in the I/O bank (e.g., it is not in the FPGA/package you chose), you cannot use an I/O standard that requires it. Instead choose a different I/O standard for the GPIO or a resource in a different I/O bank that has a VREF pin bonded out. .

gpio_rule_io_standard_stl (warning)

Message	Skip checking Vref requirement on a single-ended input configuration: input path is not used
To fix	You get this warning when the GPIO is in inout mode but you have not specified an input pin name.

gpio_rule_io_standard_differential (error)

Message	GPIO resource, <name> with differential I/O standard was configured for multiple use Differential I/O standard can only be assigned to pad P resource
To fix	You cannot use differential HSTL/SSTL for the N resource. Change the resource to a P one.

Message	Differential I/O standard is not valid on GPIO resource <name> due to the corresponding N resource not bonded
To fix	To use a differential I/O standard, both the N and P resources must be available. If the N resource is not bonded out, then you cannot use a differential standard. Choose another pair of N and P resources.

gpio_rule_dynamic_delay (error)

Message	Input dynamic delay is not supported in non-P resource, <resname>
To fix	The HSIO N resource does not support dynamic delay. Either change the delay to static or pick a P resource.

Message	Clock pin name in input dynamic delay is empty
To fix	When using the dynamic delay, you need to specify a clock input pin name.

gpio_rule_serialization (error)

Message	Register Option has to be set to register when using serialization
To fix	You cannot use serialization unless the **Register Option** is set to **register**. Either change the option or do not use serialization.

Message	Serialization cannot be used with DDIO Option
To fix	If you are using serialization, you cannot also use DDIO. Either disable serialization or set **Double Data I/O Option** to **none**.

Message	Serial and parallel clock names are required to be non-empty with serialization
To fix	You need to specify the parallel and serial clock names if you are using serialization.

Message	Serialization is not supported due to PLL is not available
To fix	When the GPIO is using the serializer, the serial and parallel clock signals must come from a PLL. You need to make two PLL clock outputs available to the GPIO.

Message	Serial and parallel clocks cannot be the same clock
To fix	Use different PLL output clocks for the signals.

Message	Serial and parallel clock names are not PLL output clocks Serial clock name is not a PLL output clock Parallel clock name is not a PLL output clock
To fix	When the GPIO is using the serializer, the serial and parallel clock signals must come from a PLL. You need to use two PLL clock outputs, one for serial and one for parallel.

Message	Serial and parallel clocks are not from the same PLL instance
To fix	Use PLL output clocks from the same PLL for the serial and parallel clocks.

Message	One of the clock frequencies is 0
To fix	Change the clock frequency to be something other than zero.

Message	Serial clock frequency has to be 4 times faster than parallel clock
To fix	Change the PLL output clock frequencies such that the serial clock is 4 times faster than the parallel clock.

Message	Invalid phase shift difference: <phase shift difference> = Serial: <serial clk shifted time> - Parallel: <parallel clk shifted time> (max=<max shift difference allowed>, min=<min shift difference allowed>)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the range of 45 to 135 degrees, relative to the phase-shifted time of the serial clock.

gpio_rule_deserialization (error)

Message	Register Option has to be enabled when using deserialization
To fix	You cannot use deserialization unless the **Register Option** is set to **register**. Either change the option or do not use deserialization.

Message	Deserialization canot be used with DDIO Option
To fix	If you are using serialization, you cannot also use DDIO. Either disable deserialization or set **Double Data I/O Option** to **none**.

Message	Serial and parallel clock names are required to be non-empty with deserialization
To fix	You need to specify the parallel and serial clock names if you are using deserialization.

Message	Deserialization is not supported due to PLL is not available
To fix	When the GPIO is using the deserializer, the serial and parallel clock signals must come from a PLL. You need to make two PLL clock outputs available to the GPIO.

Message	Serial and parallel clocks cannot be the same clock
To fix	Use different PLL output clocks for the signals.

Message	Serial and parallel clock names are not PLL output clocks Serial clock name is not a PLL output clock Parallel clock name is not a PLL output clock
To fix	When the GPIO is using the deserializer, the serial and parallel clock signals must come from a PLL. You need to use two PLL clock outputs, one for serial and one for parallel.

Message	Serial and parallel clocks are not from the same PLL instance
To fix	Use PLL output clocks from the same PLL for the serial and parallel clocks.

Message	One of the clock frequencies is 0
To fix	Change the clock frequency to be something other than zero.

Message	Serial clock frequency has to be 4 times faster than parallel clock
To fix	Change the PLL output clock frequencies such that the serial clock is 4 times faster than the parallel clock.

Message	Invalid phase shift difference: <phase shift difference> = Serial: <serial clk shifted time> - Parallel: <parallel clk shifted time> (max=<max shift difference allowed>, min=<min shift difference allowed>)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the range of 45 to 135 degrees, relative to the phase-shifted time of the serial clock.

gpio_rule_bus_hold (error)

Message	Bus hold is only supported on 1.2/1.5/1.8 V LVCMOS I/O standard HSIO only
To fix	You cannot use the bus hold option for the I/O standard you chose. Either turn off **Enable Bus Hold** or use a different I/O standard.

gpio_rule_differential_stl_inout (error)

Message	For inout mode with differential STL, output enable (N) pin name must be configured
To fix	Specify the output enable pin name.

gpio_rule_cfg_io_standard_valid (error)

Message	Unsupported I/O standard
To fix	The I/O standard you chose is not supported. Choose another one. Refer to Types of GPIO on page 60.

gpio_rule_cfg_slew_rate (error)

Message	Resource <name> does not support Slew Rate feature. It will be ignored
To fix	The Titanium HVIO pins do not support slew rate. Either diable that option or choose another resource.

gpio_rule_cfg_dyn_delay (error)

Message	Resource <name> does not support Dynamic Delay feature.
To fix	The resource you used does not support dynamic delay. Refer to Features for HVIO and HSIO Configured as GPIO on page 62 for which GPIO support that feature. Either turn off dynamic delay or choose another resource that supports it.

gpio_rule_cfg_bus_hold (warning)

Message	Resource <name> does not support Bus Hold feature. It will be ignored
To fix	Not all resources support bus hold. Refer to Features for HVIO and HSIO Configured as GPIO on page 62 for which GPIO support that feature. Either turn off bus hold or choose another resource that supports it.

gpio_rule_cfg_dyn_pullup (warning)

Message	Resource <name> does not support Dynamic Pullup feature. It will be ignored
To fix	Not all resources support dynamic pullup. Refer to Features for HVIO and HSIO Configured as GPIO on page 62 for which GPIO support that feature. Change the **Pull Option** or choose another resource that supports it.

gpio_rule_cfg_serialization (error)

Message	Resource <name> does not support Serialization feature.
To fix	Not all resources support serialization. Refer to Features for HVIO and HSIO Configured as GPIO on page 62 for which GPIO support that feature. Turn off serialization or choose another resource that supports it.

gpio_rule_cfg_deserialization (error)

Message	Resource <name> does not support Deserialization feature.
To fix	Not all resources support deserialization. Refer to Features for HVIO and HSIO Configured as GPIO on page 62 for which GPIO support that feature. Turn off deserialization or choose another resource that supports it.

Chapter 6

LVDS Interface

Contents:

Each HSIO block can use a pair of I/O pins as an LVDS receiver (RX), transmitter (TX), or bidirectional (RX/TX) signal.

HSIO Configured as LVDS

You can configure each HSIO block in RX, TX, or bidirectional LVDS mode. As LVDS, the HSIO has these features:

- Programmable V_{OD}, depending on the I/O standard used.
- Programmable pre-emphasis.
- Up to 1.5 Gbps.
- Programmable 100 Ω termination to save power (you can enable or disable it at runtime).
- LVDS input enable to dynamically enable/disable the LVDS input.
- Support for full rate or half rate serialization.
- Up to 10-bit serialization to support protocols such as 8b10b encoding.
- Programmable delay chains.
- Optional 8-word FIFO for crossing from the parallel (slow) clock to the user's core clock to help close timing (RX only).
- Dynamic phase alignment (DPA) that automatically eliminates skew for clock to data channels and data to data channels by adjusting a delay chain setting so that data is sampled at the center of the bit period. The DPA supports full-rate serialization mode only.

Table 43: Full and Half Rate Serialization

Mode	Description	Example
Full rate clock	In full rate mode, the fast clock runs at the same frequency as the data and captures data on the positive clock edge.	Data rate: 800 Mbps Serialization/Deserialization factor: 8 Slow clock frequency: 100 Mhz (800 Mbps / 8) Fast clock frequency: 800 Mhz
Half rate clock	In half rate mode, the fast clock runs at half the speed of the data and captures data on both clock edges.	Data rate: 800 Mbps Serialization / Deserialization factor: 8 Slow clock frequency: 100 Mhz (800 Mbps / 8) Fast clock frequency: 400 Mhz (800 / 2)

You use a PLL to generate the serial (fast) and parallel (slow) clocks for the LVDS pins. The slow clock runs at the data rate divided by the serialization factor.

LVDS RX

You can configure an HSIO block as one LVDS RX signal.

Figure 29: LVDS RX Interface Block Diagram

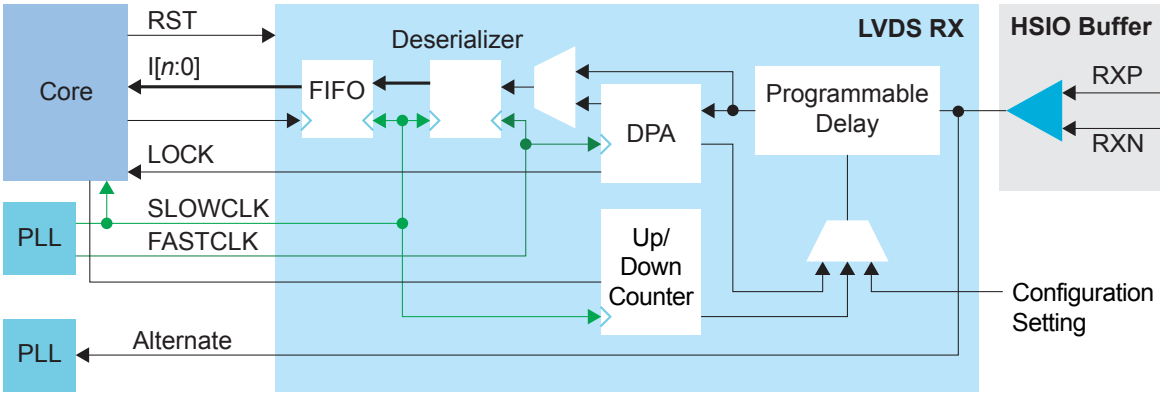
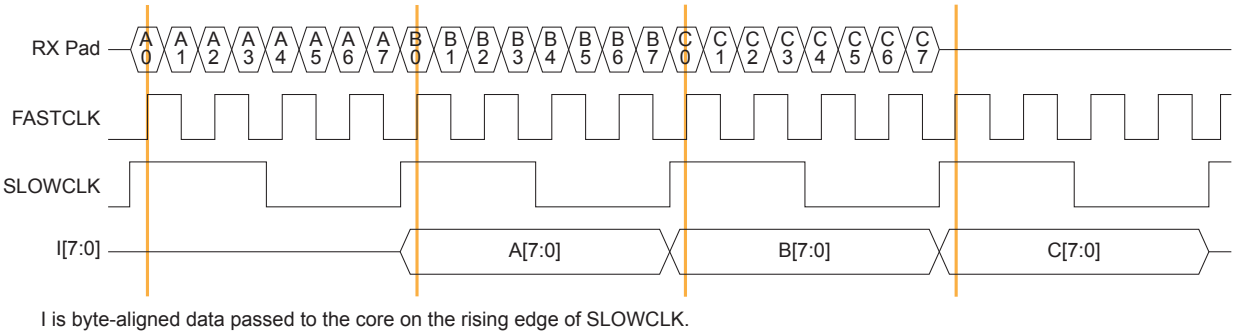


Table 44: LVDS RX Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Description
I[9:0]	Output	SLOWCLK	Parallel input data to the core. The width is programmable.
ALT	Output		Alternate input, only available for an LVDS RX resource in bypass mode (deserialization width is 1; alternate connection type). Alternate connections are PLL_CLKIN, PLL_EXTFB, GCLK, and RCLK.
SLOWCLK	Input	–	Parallel (slow) clock.
FASTCLK	Input	–	Serial (fast) clock.
FIFO_EMPTY	Output	FIFOCLK	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Indicates that the FIFO is empty.
FIFOCLK	Input	–	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Core clock to read from the FIFO.
FIFO_RD	Input	FIFOCLK	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Enables FIFO to read.
RST	Input	FIFOCLK SLOWCLK	(Optional) This signal is available when deserialization is enabled. Asynchronous. Resets the FIFO and deserializer. If the FIFO is enabled, it is relative to FIFOCLK; otherwise it is relative to SLOWCLK.
ENA	Input	–	Dynamically enable or disable the LVDS input buffer. Can save power when disabled. 1: Enabled 0: Disabled
TERM	Input	–	The signal is available when dynamic termination is enabled. Enables or disables termination in dynamic termination mode. 1: Enabled 0: Disabled
LOCK	Output		(Optional) This signal is available when you set **Delay Mode** to **dpa**. Indicates that the DPA has achieved training lock and data can be passed.
DLY_ENA	Input	SLOWCLK	This signal is required when you set **Delay Mode** to **dynamic** or **dpa**. Enable the dynamic delay control or the DPA circuit, depending on the LVDS RX delay settings.
DLY_INC	Input	SLOWCLK	This signal is required when you set **Delay Mode** to **dynamic**. Dynamic delay control. Cannot be used with DPA enabled. When DLY_ENA is 1: 1: Increments 0: Decrements
DLY_RST	Input	SLOWCLK	(Optional) This signal is available when you set **Delay Mode** to **dpa** or **dynamic**. Reset the delay counter or the DPA circuit, depending on the LVDS RX delay settings.
DBG[5:0]	Output	SLOWCLK	DPA debug pin. Outputs the final delay chain settings when DPA achieved lock.

The following waveform shows the relationship between the fast clock, slow clock, RX data coming in from the pad, and byte-aligned data to the core.

Figure 30: LVDS RX Timing Example Serialization Width of 8 (Half Rate)



Note: For LVDS RX interfaces with multiple LVDS RX lanes and an LVDS RX clock input, use the LVDS RX blocks from the same side of the FPGA to minimize skew between data lanes and RX clock input.

LVDS TX

You can configure an HSIO block as one LVDS TX signal. LVDS TX can be used in the serial data output mode or reference clock output mode.

Figure 31: LVDS TX Interface Block Diagram

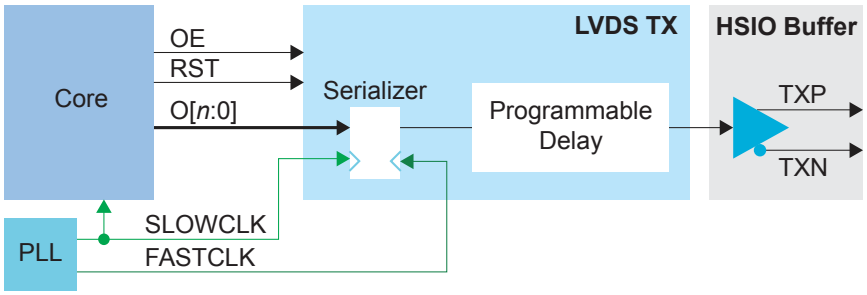
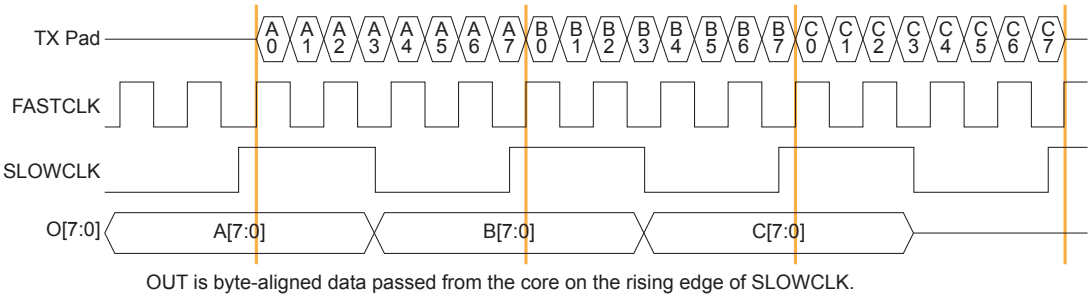


Table 45: LVDS TX Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Description
O[9:0]	Input	SLOWCLK	Parallel output data from the core. The width is programmable.
SLOWCLK	Input	–	Parallel (slow) clock.
FASTCLK	Input	–	Serial (fast) clock.
RST	Input	SLOWCLK	(Optional) This signal is available when serialization is enabled. Resets the serializer.
OE	Input	–	(Optional) Output enable signal.

The following waveform shows the relationship between the fast clock, slow clock, TX data going to the pad, and byte-aligned data from the core.

Figure 32: LVDS Timing Example Serialization Width of 8 (Half Rate)



OUT is byte-aligned data passed from the core on the rising edge of SLOWCLK.

Note: For LVDS TX interfaces with multiple LVDS TX lanes and an LVDS TX reference clock output, use the LVDS TX blocks from the same side of the FPGA to minimize skew between data lanes and TX reference clock output.

LVDS Bidirectional

You can configure an HSIO block as one LVDS bidirectional signal. You must use the same serialization for the RX and TX.

Figure 33: LVDS Bidirectional Interface Block Diagram

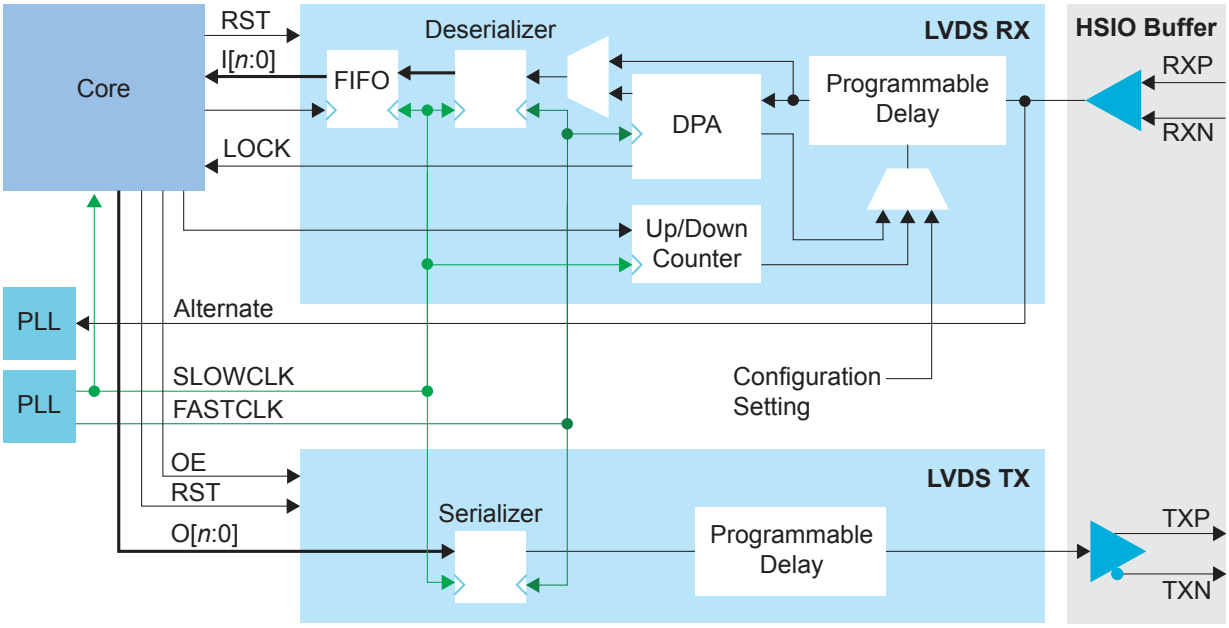


Table 46: LVDS Bidirectional Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Description
I[9:0]	Output	SLOWCLK	Parallel input data to the core. The width is programmable.
INSLOWCLK	Input	–	Parallel (slow) clock for RX.
INFASTCLK	Input	–	Serial (fast) clock for RX.
FIFO_EMPTY	Output	FIFOCLK	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Indicates that the FIFO is empty.
FIFOCLK	Input	–	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Core clock to read from the FIFO.
FIFO_RD	Input	FIFOCLK	This signal is required when you turn on the **Enable Clock Crossing FIFO** option. Enables FIFO to read.
INRST	Input	FIFOCLK SLOWCLK	This signal is available when deserialization is enabled. Asynchronous. Resets the FIFO and RX deserializer. If the FIFO is enabled, it is relative to FIFOCLK; otherwise it is relative to SLOWCLK.
ENA	Input	–	Dynamically enable or disable the LVDS input buffer. Can save power when disabled. 1: Enabled 0: Disabled
TERM	Input	–	The signal is available when dynamic termination is enabled. Enables or disables termination in dynamic termination mode. 1: Enabled 0: Disabled

Signal	Direction	Clock Domain	Description
LOCK	Output		(Optional) This signal is available when you set **Delay Mode** to **dpa**. Indicates that the DPA has achieved training lock and data can be passed.
DLY_ENA	Input	SLOWCLK	This signal is required when you set **Delay Mode** to **dynamic** or **dpa**. Enable the dynamic delay control or the DPA circuit, depending on the bidirectional LVDS delay settings.
DLY_INC	Input	SLOWCLK	This signal is required when you set **Delay Mode** to **dynamic**. Dynamic delay control. Cannot be used with DPA enabled. When DLY_ENA is 1, 1: Increments 0: Decrements
DLY_RST	Input	SLOWCLK	(Optional) This signal is available when you set **Delay Mode** to **dpa** or **dynamic**. Reset the delay counter or the DPA circuit, depending on the bidirectional LVDS delay settings.
DBG[5:0]	Output	SLOWCLK	DPA debug pin. Outputs the final delay chain settings when DPA achieved lock.
O[9:0]	Input	SLOWCLK	Parallel output data from the core. The width is programmable.
OUTSLOWCLK	Input	–	Parallel (slow) clock for TX.
OUTFASTCLK	Input	–	Serial (fast) clock for TX.
OUTRST	Input	SLOWCLK	This signal is available when serialization is enabled. Resets the TX serializer.
OE	Input	–	Output enable signal.

LVDS Pads

Table 47: LVDS Pads

Signal	Direction	Description
P	Output	Differential pad P.
N	Output	Differential pad N.

Using the LVDS Block

The LVDS block defines the functionality of the LVDS pins. You can choose whether the block is a transmitter (TX), receiver (RX), or bidirectional.

LVDS TX

Table 48: LVDS TX Options

Option	Choices	Description
Instance Name	User defined	Enter a name.
LVDS Resource	Resource list	Choose a resource.
Output Differential Type	lvds	Use for LVDS, RSDS, and mini-LVDS.
	sublvds	Use for subLVDS.
	slvs	Use for SLVS.
	custom	Choose this option when you want to use a VREF pin to specify the differential. Set the GPIO input connection type to **vref**. Choose a GPIO pin with that supports VREF in the same bank as the LVDS TX resource.
Output Differential, VOD	Typical, large, small	The actual voltage depends on this setting and the diffential type and is shown in the Block Summary **Value** field.
Output-Pre-Emphasis	high, medium-high, medium-low, low	Choose an output pre-emphasis setting.
Mode	serial data output	Use the transmitter as a simple output buffer or serialized output.
	reference clock output	Use the transmitter as a clock output. Specify the serial and parallel clocks. When choosing this mode, the serialization width should match the serialization for the rest of the LVDS bus.
Output Pin/Bus Name	User defined	Output pin or bus that feeds the LVDS transmitter parallel data. The width should match the serialization factor.
Output Enable Pin Name	User defined	Use with serial data output mode.
Enable Serialization	Off	Use as a simple buffer.
	On	Use as an LVDS serializer: • Optionally enable half rate serialization. • Choose a value of 2, 3, 4, 5, 6, 7, 8, or 10 (1 is a simple buffer). A value of 9 is not legal. • Specify the serial clock and parallel clock. • Specify reset pin name.
Static Mode Delay Setting	0 - 63	Choose the amount of static delay, each step adds approximately 25 ps of delay.

The maximum LVDS rate is 1.5 Gbps.

- *Half rate calculation*—serial clock frequency = parallel clock frequency * (serialization / 2)
- *Full rate calculation*—serial clock frequency = parallel clock * serialization

The serial clock must have a phase shift that is between 45 degrees and 135 degrees. Both clocks must come from the same PLL.

The serial clock (also known as the fast clock) outputs data to the pin, the parallel clock (also known as the slow clock) transfers it from the core. An equation defines the relationship between the clocks. For LVDS TX the parallel clock captures data from the core and the serial clock outputs it to the LVDS buffer.

In half-rate mode, new data is output on both edges of the serial clock, in full rate mode it is only on the rising (positive) edge.

LVDS RX

Table 49: LVDS RX Options

Option	Choices	Description
Instance Name	User defined	Enter a name.
LVDS Resource	Resource list	Choose a resource.
Connection Type	normal	LVDS RX function.
	pll_clkin	Alternate function. Use as PLL reference clock.
	pll_extfb	Alternate function. Use as PLL external feedback.
	gclk	Alternate function. Use as global clock.
	rclk	Alternate function. Use as regional clock.
Input Pin/Bus Name	User defined	Input pin or bus that feeds the LVDS transmitter parallel data. The width should match the deserialization factor.
Dynamic Enable Pin Name	User defined	Dynamically enables or disables the LVDS RX buffer. Disabling the buffer can reduce power consumption when the pin is not in use.
Enable Common Mode Driver	On, off	If you implement an AC coupled connection, turn on this option. For a typical DC coupled connection, leave this option off.
Enable SLVS	On, off	Turn on to use SLVS instead of LVDS.
Termination	on, off, dynamic	For **dynamic**, specify the pin that controls the dynamic termination.
Enable Deserialization	Off	Use as a simple buffer.
	On	Use as an LVDS deserializer: • Optionally enable half rate serialization. • Choose a width of 2, 3, 4, 5, 6, 7, 8, or 10 (1 is a simple buffer). A width of 9 is not legal. • Specify the serial clock and parallel clock. • Optionally turn on **Enable Clock Crossing FIFO**, which uses a FIFO to cross between the slow clock and the user core clock. Specify the FIFO read, clock, and empty pin names.
Delay Mode	static	Integer from 0 - 63. Each step adds approximately 25 ps of delay.
	dynamic	Specify the pin names to control the dynamic delay.
	dpa	Dynamic phase alignment automatically sets the delay value.

The serial clock (also known as the fast clock) captures data from the pin, the parallel clock (also known as the slow clock) transfers it to the core. An equation defines the relationship between the clocks.

The maximum LVDS rate is 1.5 Gbps.

- *Half rate calculation*—serial clock frequency = parallel clock frequency * (deserialization / 2)
- *Full rate calculation*—serial clock frequency = parallel clock * deserialization

The serial clock must have a phase shift that is between 45 and 135 degrees. Both clocks must come from the same PLL.

LVDS Bidirectional

The LVDS bidirectional block has the same options and choices as the LVDS RX and TX blocks.

Important: You must use the same value for the serialization/deserialization.

PLL Requirements for Serial and Parallel Clocks

With Titanium FPGAs, you need to use the output clocks from specific PLLs as the LVDS serial and parallel clocks.

Table 50: PLL Requirements

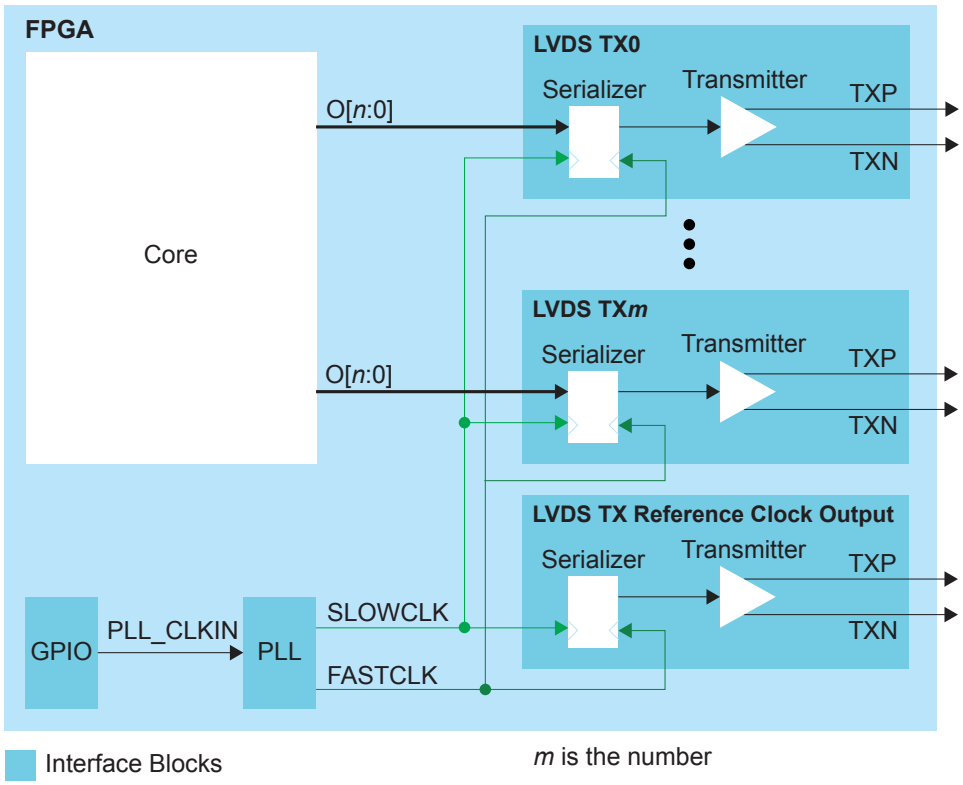
FPGA	Side	PLL
Ti35, Ti60, Ti60ES	Left	BL_PLL, TL_PLL
	Right	BR_PLL, TR_PLL
	Top	TR_PLL, TL_PLL
	Bottom	BR_PLL, BL_PLL

Learn more: Refer to the AN 044: Aligning LVDS Clock and Data for more information about aligning the clock and data for LVDS interface in different applications.

Create an LVDS TX Interface

The following figure shows a completed LVDS TX interface, where *n* is the serialization width and *m* is the number of TX lanes.

Figure 34: Complete LVDS TX Interface Block Diagram



Note: Use LVDS TX blocks from the same side of the FPGA to minimize skew between data lanes and TX reference clock output in an LVDS TX interface.

Follow these steps to build an LVDS TX interface using the Efinity® Interface Designer.

1. Add a PLL block with the following settings:

Option	Description
Resource	You can use any PLL resource.
Reference Clock Mode	External
Reference Clock Frequency	Any
Output Clock	For LVDS serializer widths 2 - 8, define the output clocks so that you have one for the fast clock (serial) and one for the slow clock (parallel). Set the relationship between the clocks such that the serial clock frequency = parallel clock frequency * (serialization / 2). The serial clock must use the 90 degree phase shift.

2. Add a GPIO block with these settings to provide the reference clock input to the PLL:

Option	Description
Mode	Input
Pin Name	Any
Connection Type	pll_clkin
GPIO Resource	Assign the dedicated PLL_CLKIN pin that corresponds to the PLL you chose.

3. Add an LVDS TX block with these settings:

Option	Description
LVDS Type	Transmitter (TX)
LVDS Resource	Any channel
Mode	Serial data output
Enable Serialization	On
Serialization Width	*n*
Output Pin/ Bus Name	Any
Serial Clock Pin Name	Use the fast clock output name that corresponds to the PLL you chose.
Parallel Clock Pin Name	Use the slow clock output name that corresponds to the PLL you chose.

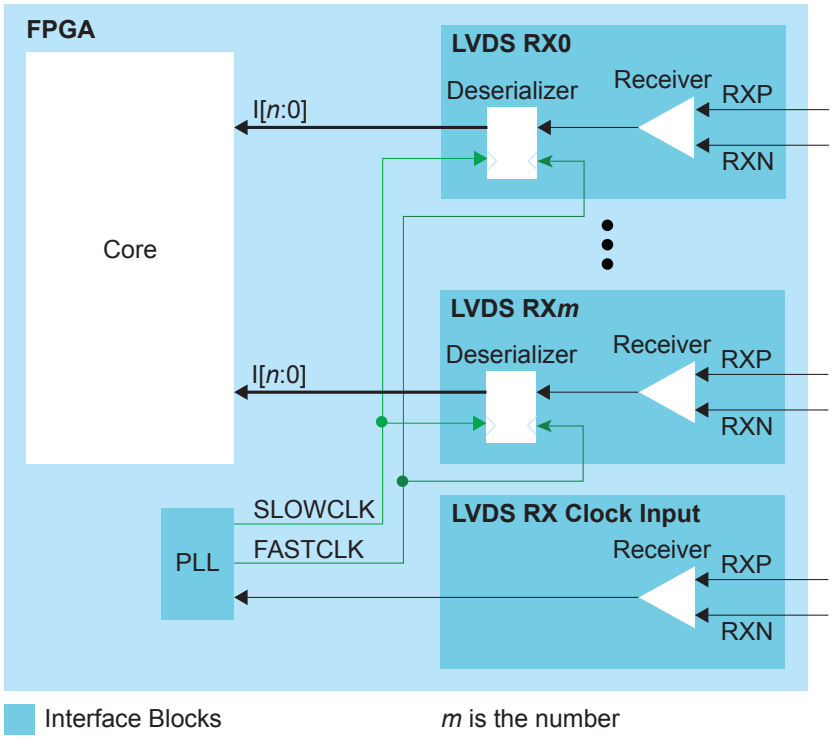
4. Repeat step 3 for each LVDS TX data lane you want to implement.
5. Add another LVDS block that will serve as the LVDS TX reference clock output:

Option	Description
LVDS type	Transmitter (TX)
LVDS resource	Any channel
Mode	Reference clock output
Enable Serialization	On
Serialization width	*n*
Output pin/ bus name	Any
Parallel clock division	1: The output clock from the LVDS TX lane is parallel clock frequency. 2: The output clock from the TX lane is half of the parallel clock frequency.
Serial clock pin name	Specify the fast clock output name that corresponds to the PLL you chose.
Parallel clock pin name	Use the slow clock output name that corresponds to the PLL you chose.

Create an LVDS RX Interface

The following figure shows a completed LVDS RX interface, where *n* is the deserialization width and *m* is the number of RX lanes.

Figure 35: Complete LVDS RX Interface Block Diagram



Note: Use LVDS RX blocks from the same side of the FPGA to minimize skew between data lanes and RX clock input in an LVDS RX interface.

Follow these steps to build an LVDS RX interface using the Efinity® Interface Designer.

1. Add an LVDS / SLVS RX block to act as the PLL reference clock input:

Option	Description
LVDS Type	Receiver (RX)
LVDS Resource	Use any LVDS resource.
Connection Type	pll_clkin
Input Pin/Bus Name	Use the clock LVDS RX clock output name as the incoming clock.

2. Add a PLL block with the following settings:

Option	Description
Resource	Use any PLL resource.
Reference Clock Mode	External
Reference Clock Frequency	Set the reference clock frequency to match the clock coming from the LVDS RX reference clock you created in step 1.
Output Clock	For LVDS deserializer widths 2 - 8, define the output clocks so that you have one for the fast clock (serial) and one for the slow clock (parallel). Set the relationship between the clocks such that the serial clock frequency = parallel clock frequency * (serialization / 2). The serial clock must use the 90 degree phase shift.

3. Add an LVDS RX block with these settings:

Option	Description
LVDS Type	Receiver (RX)
LVDS Resource	Any channel
Enable Deserialization	On
Deserialization Width	*n*
Output Pin/ Bus Name	Any
Serial Clock Pin Name	Use the fast clock output name that corresponds to the PLL you chose.
Parallel Clock Pin Name	Use the slow clock output name that corresponds to the PLL you chose.

4. Repeat step 3 for each LVDS RX data lane you want to implement.

Design Check: LVDS Messages

When you check your design, the Interface Designer applies design rules to your LVDS settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

lvds_rule_bidir_tx (error)

Message	Output enable pin name must be configured in Bidirectional LVDS Tx
To fix	If you are using a bidirectional LVDS block, you need to specify the output enable pin name.

lvds_rule_clkout_mode (error)

Message	Serial clock name must be configured in clock output mode
To fix	When you are using the LVDS serializer (serialization width greater than 1), you need to specify the serial clock pin name.

Message	Parallel clock name must be configured in clock output mode
To fix	When you are using the LVDS serializer (serialization width greater than 1), you need to specify the parallel clock pin name.

Message	Clock output mode is not supported with serialization width 1
To fix	When you set the serialization width to 1, you are bypassing the serializer and using the block as a simple buffer. As a simple buffer, you cannot use the block as a reference clock output. Change the serialization width.

Message	Clock output mode is not supported with serialization disabled
To fix	If you turn off **Enable Serialization**, you are bypassing the serializer and using the block as a simple buffer. As a simple buffer, you cannot use the block as a reference clock output. Turn on **Enable Serialization** and set the serialization width.

lvds_rule_deserial_rate (error)

Message	Half rate deserialization only allowed with even deserialization width
To fix	When you turn on **Enable Half Rate Serialization**, you can only use an even number for the deserialization width. Change the width to an even number or turn the option off.

lvds_rule_output_mode (error)

Message	Instance of LVDS Tx has invalid configuration
To fix	The block settings are not correct. It might be best to remove the block and start over :)

Message	Output name must be configured in data output mode
To fix	Specify a valid pin name.

Message	Parallel clock name must be configured in data output mode
To fix	When you are using the LVDS serializer (serialization width greater than 1), you need to specify the parallel clock pin name.

Message	Serial clock name must be configured in data output mode
To fix	When you are using the LVDS serializer (serialization width greater than 1), you need to specify the serial clock pin name.

lvds_rule_resource (error)

Message	Resource name is empty Resource <string> is not a valid LVDS device instance
To fix	You need to choose a valid resource.

lvds_rule_usage (error)

Message	Resource <res name> was assigned multiple times
To fix	You cannot assign the same resource to more than one block type. Change the resource to a different one.

lvds_rule_rx_alt_conn (error)

Message	Connection type <type> is not supported by the resource
To fix	If you want to use the alternate funciton of an LVDS block, you need to choose a resource that supports it. You can filter for resources by alternate function in the Resource Assigner.

Message	The resource only supports normal connection type
To fix	You need to choose the normal connection type or assign a different resource that supports the connection type you want to use. You can filter for resources by alternate function in the Resource Assigner.

lvds_rule_alt_conn (warning)

Message	Connection type <type> must be used by valid PLL
To fix	The LVDS block is connected to a PLL clock input but is the resource you assigned does not support the pll_clkin alternate function. Choose a different resource that supports it. You can filter resources by alternate function in the Resource Assigner.

Message	Connection type <type> cannot be used on an unbonded resource
To fix	You get this error if the resource you choose is not available in the FPGA/package combination you are using. Choose another resource.

Message	pll_clkin connection to PLL clock source not being used in <instance>
To fix	The LVDS block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. Make sure that the clock you are choosing in the PLL is associated with this GPIO's resource.

Message	pll_clkin connection to PLL clock source but none of the external clock source in PLL <instance> is selected
To fix	The LVDS block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. In the PLL block, choose external or dynamic as the Clock Source and make sure that the clock you are choosing is associated with this GPIO's resource.

Message	pll_clkin connection to PLL clock source but PLL Clock source on <instance> is set to core
To fix	The LVDS block is set to be a PLL reference clock (pll_clkin connection type) but the PLL is not configured to use it. In the PLL block, choose external or dynamic as the Clock Source and make sure that the clock you are choosing is associated with this GPIO's resource.

Message	pll_extfb connection to PLL external feedback pin but PLL feedback on <inst> is not set to default
To fix	The LVDS block is set to be external feedback for the PLL (pll_extfb connection type) but the PLL is not configured to use it. In the PLL Clock Calculator, choose External as the Feedback Mode.

lvds_rule_rx_clock (error)

Message	Serial and parallel clocks cannot be the same clock
To fix	You cannot use the same clock for both the serial (FASTCLK) and parallel (SLOWCLK) clocks.

Message	Serial clock name is not a PLL output clock
To fix	Use a PLL output clock as the serial (FASTCLK) clock.

Message	Parallel clock name is not a PLL output clock
To fix	Use a PLL output as the parallel (SLOWCLK) clock.

Message	Serial and parallel clocks are not from the same PLL instance
To fix	You need to use the same PLL to generate both clocks.

Message	Invalid phase shift difference: <phase shift difference> = Serial: <serial clk shifted time> - Parallel: <parallel clk shifted time> (max=<max shift difference allowed>, min=<min shift difference allowed>)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the range of 45 to 135 degrees, relative to the phase-shifted time of the serial clock.

Message	Serial clock frequency has to be <float> times faster than parallel clock
To fix	Make sure that the PLL output clock frequencies are set correctly. *Half rate calculation*—serial clock frequency = parallel clock frequency * (serialization / 2) *Full rate calculation*—serial clock frequency = parallel clock * serialization

Message	Invalid phase shift difference: {phase shift difference} = Serial: {serial clk shifted time} - Parallel: {parallel clk shifted time} (max={max shift difference allowed}, min={min shift difference allowed) Example: Invalid phase shift difference: 0.0000 ps = Serial: 400.0000 ps = Parallel: 400.0000 ps (max=600.0000 ps, min = 200.0000 ps)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the specified range.

lvds_rule_rx_clock_region (error)

Message	Serial and Parallel clocks generated by PLL have to be driven to the same clock network. <Serial|Parallel> clock <name> was generated by PLL output clock 4 that connects to regional clock network
To fix	In Ti35, Ti60 FPGAs, the PLL's output clock 4 can only drive the regional clock network. You should use the other clock outputs for the serial and parallel clocks.

lvds_rule_rx_config (error)

Message	Input name must be configured
To fix	Specify a valid pin name.

Message	Serial clock name must be configured
To fix	When you are using the LVDS deserializer (deserialization width greater than 1), you need to specify the serial clock pin name.

Message	Parallel clock name must be configured
To fix	When you are using the LVDS deserializer (deserialization width greater than 1), you need to specify the parallel clock pin name.

lvds_rule_rx_distance (error)

Message	These HSIO GPIO must be placed at least 1 pair away from LVDS <name> in order to avoid noise coupling from GPIO to LVDS: <violated list>
To fix	When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO used as LVDS RX in the same bank. This separation reduces noise.

lvds_rule_rx_dpa (error)

Message	Half-rate deserialization is not supported with DPA delay mode
To fix	You can only use full-rate serialization with DPA mode. Turn off the **Enable Half Rate Deserialization** option.

lvds_rule_rx_dpa_es_device (error)

Message	DPA delay mode is not supported in ES device
To fix	The ES FPGA does not support DPA.

lvds_rule_rx_dpa_data_rate (info)

Message	DPA usage on data rate < 800Mbps is not significant. DPA might not find an optimum step to lock, dis-enable DPA and assume DPA has locked when DLY_DBG stepped on 0 or 63. Refer to AN 044 for more info.
To fix	Data rates below 800 Mbps can result in a rolloff or overflow in the value of DLY_DBG, which in turn can cause a jump directly from 0 to 63, or 63 to 0. Such a rolloff invalidates the DPA operation. Refer to "Calibrating with Dynamic Phase Alignment (DPA) (Titanium/Topaz only)" in **AN 044: Aligning LVDS Clock and Data Signals**.

lvds_rule_rx_dpa_serial (error)

Message	DPA delay mode is not supported with deserialization disabled
To fix	You cannot use dynamic phase alignment in bypass mode.

Message	DPA delay mode is not supported with deserialization width less than 3
To fix	You cannot use dynamic phase alignment with x1 or x2 modes.

lvds_rule_rx_empty_pins (error)

Message	Empty pin names found: <list>
To fix	You need to specify the pin names listed in the message.

lvds_rule_rx_fifo (error)

Message	Clock Crossing FIFO is not supported with deserialization width <1/2>
To fix	The Clock Crossing FIFO is only available for deserialization widths > 2. Disable the Clock Crossing FIFO or change the serialization value.

Message	Clock Crossing FIFO is only supported with deserialization enabled
To fix	The Clock Crossing FIFO is only available for deserialization widths > 2. Disable the Clock Crossing FIFO or change the serialization value.

lvds_rule_rx_param (error)

Message	Invalid parameters configuration: <list>
To fix	One of the parameters you set was incorrect. Review any other errors for details.

lvds_rule_rx_pll_refclk (error)

Message	Serial clock name is not a PLL output clock
To fix	Use a PLL output clock as the serial (FASTCLK) clock.

Message	Parallel clock name is not a PLL output clock
To fix	Use a PLL output as the parallel (SLOWCLK) clock.

lvds_rule_rx_pll_refclk (warning)

Message	Serial clock is expected to be from the following PLL instance: <resource>
To fix	Only a specific PLL instance can drive the LVDS RX clocks. Change the PLL to use that resource. See PLL Requirements for Serial and Parallel Clocks on page 98.

Message	PLL driving the serial clock should have its reference clock from an LVDS in pll_clkin connection type
To fix	The PLL's reference clock needs to be driven by a specific resource. Create an LVDS RX block and set the **Connection Type** to **pll_clkin**. Then use that block as the PLL reference clock.

Message	Parallel clock is expected to be from the following PLL instance: {}
To fix	Only a specific PLL instance can drive the LVDS RX clocks. Change the PLL to use that resource. See PLL Requirements for Serial and Parallel Clocks on page 98.

Message	PLL driving the parallel clock should have its reference clock from an LVDS in pll_clkin connection type
To fix	The PLL's reference clock needs to be driven by a specific resource. Create an LVDS RX block and set the **Connection Type** to **pll_clkin**. Then use that block as the PLL reference clock.

lvds_rule_rx_serial_width (error)

Message	Unsupported deserializaion width: 9
To fix	The LVDS block does not support a deserialization wiudth of 9. Choose another width.

lvds_rule_tx_width_1or2 (error)

Message	Parallel clock name is required with serialization width 2 Serialization width <1/2> only requires the parallel clock name to be specified
To fix	When you are using the LVDS serializer (serialization width greater than 1), you need to specify the parallel clock pin name.

lvds_rule_rx_width_1or2 (error)

Message	Parallel clock name is required with deserialization width 2 Deserialization width <1/2> only requires the parallel clock name to be specified
To fix	When you are using the LVDS deserializer (serialization width greater than 1), you need to specify the parallel clock pin name.

lvds_rule_serial_rate (error)

Message	Half rate serialization only allowed with even serialization width
To fix	When you turn on **Enable Half Rate Serialization**, you can only use an even number for the serialization width. Change the width to an even number or turn the option off.

lvds_rule_tx_clock (error)

Message	Serial and parallel clocks cannot be the same clock
To fix	You cannot use the same clock for both the serial (FASTCLK) and parallel (SLOWCLK) clocks.

Message	Serial clock name is not a PLL output clock
To fix	Use a PLL output clock as the serial (FASTCLK) clock.

Message	Parallel clock name is not a PLL output clock
To fix	Use a PLL output as the parallel (SLOWCLK) clock.

Message	Serial and parallel clocks are not from the same PLL instance
To fix	You need to use the same PLL to generate both clocks.

Message	Invalid phase shift difference: <phase shift difference> = Serial: <serial clk shifted time> - Parallel: <parallel clk shifted time> (max=<max shift difference allowed>, min=<min shift difference allowed>)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the range of 45 to 135 degrees, relative to the phase-shifted time of the serial clock.

Message	Serial clock frequency has to be <float> times faster than parallel clock
To fix	Make sure that the PLL output clock frequencies are set correctly. *Half rate calculation*—serial clock frequency = parallel clock frequency * (serialization / 2) *Full rate calculation*—serial clock frequency = parallel clock * serialization

Message	Invalid phase shift difference: {phase shift difference} = Serial: {serial clk shifted time} - Parallel: {parallel clk shifted time} (max={max shift difference allowed}, min={min shift difference allowed) Example: Invalid phase shift difference: 0.0000 ps = Serial: 400.0000 ps = Parallel: 400.0000 ps (max=600.0000 ps, min = 200.0000 ps)
To fix	Adjust the phase shift for the serial clock and parallel clock to ensure that the phase-shifted time difference falls within the specified range.

lvds_rule_tx_clock_region (error)

Message	Serial and Parallel clocks generated by PLL have to be driven to the same clock network. <Serial | Parallel> clock <name> was generated by PLL output clock 4 that connects to regional clock network
To fix	In Ti35, Ti60 FPGAs, the PLL's output clock 4 can only drive the regional clock network. You should use the other clock outputs for the serial and parallel clocks.

lvds_rule_tx_distance (error)

Message	These HSIO GPIO must be placed at least 1 pair away from LVDS <name> in order to avoid noise coupling from GPIO to LVDS: <violated list>
To fix	When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO used as LVDS TX in the same bank. This separation reduces noise.

lvds_rule_tx_empty_pins (error)

Message	Empty pin names found: <list>
To fix	You need to specify the pin names listed in the message.

lvds_rule_tx_param (error)

Message	Invalid parameters configuration: <list>
To fix	One of the parameters you set was incorrect. Review any other errors for details.

lvds_rule_tx_serial_width (error)

Message	Unsupported serializaion width: 9
To fix	The LVDS block does not support a serialization wiudth of 9. Choose another width.

lvds_rule_tx_vref (error)

Message	This resource is reserved as vref for bank <name>. Us a different resource to configure LVDS Tx custom output differential type
To fix	Some resources can be used as the VREF for an I/O standard. If you are using an I/O standard that uses a VREF pin, you must use this resource as a VREF. Choose another resource for the LVDS function.

Message	GPIO <name> has to be configured as vref input mode to support LVDS Tx custom output differential type GPIO <name> has to be configured as vref input to support LVDS Tx custom output differential type
To fix	If you are using an I/O standard that uses a VREF pin, you must use this resource as a VREF. Configure the GPIO as an input and choose **vref** as the **Connection Type**.

Message	LVDS Tx custom output differential type cannot be used due to unbonded vref resource on the same bank <bank> LVDS Tx custom output differential type cannot be used due to vref resource not bonded out
To fix	If a VREF pin is not available in the I/O bank (e.g., it is not in the FPGA/package you chose), you cannot use an I/O standard that requires it. Instead choose a different I/O standard or a different resource.

lvds_rule_clkout_ser_disabled (error)

Message	Output clock name must be configured in clock output mode with serialization disabled
To fix	Specify the output clock name.

lvds_rule_rx_pll_feedback (warning)

Message	PLL <pll_slow_inst_name> driving the LVDS Rx clock sources should have its feedback mode set to core for optimized performance
To fix	Set feedback mode of the PLL to core for better performance.

Chapter 7

HyperRAM Interface

Contents:

The Ti35 and Ti60 FPGAs in the F100S3F2 package include an integrated HyperRAM memory. You use the Interface Designer to connect this block to your user design. Only the Ti35 or Ti60 can communicate with the on-board HyperRAM.

About the HyperRAM

The HyperRAM has a density of 256 Mbits and a clock rate of up to 200 MHz. The HyperRAM supports double-data rates of up to 400 Mbps and supports a 16 bit data bus.

Figure 36: HyperRAM Block Diagram

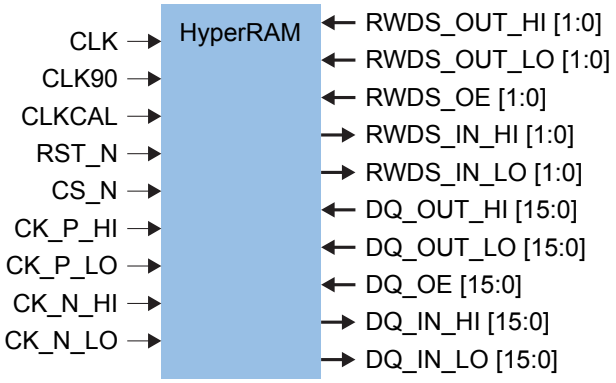


Table 51: HyperRAM Signals (Interface to FPGA Fabric)

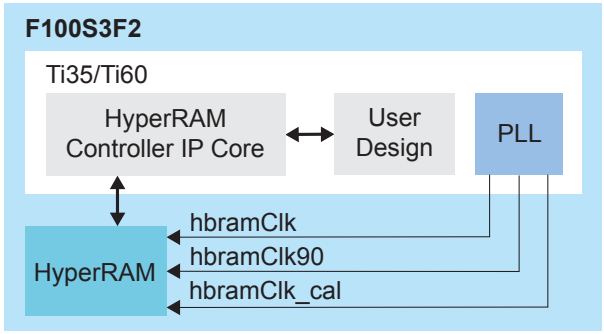
Signal	Direction	Description
CLK	Input	HyperRAM controller clock.
CLK90	Input	90 degree phase-shifted version of CLK.
CLKCAL	Input	Calibration clock for input data.
RST_N	Input	Active-low HyperRAM reset.
CS_N	Input	Active-low HyperRAM chip select signal.
CK_P_HI	Input	The clock provided to the HyperRAM. The clock is not required to be free-running. Registered in normal mode of DDIO.
CK_P_LO	Input	
CK_N_HI	Input	
CK_N_LO	Input	
RWDS_OUT_HI [1:0]	Input	Read/write data strobe input ports for data mask during write operation. Registered in normal mode/resync mode of DDIO.
RWDS_OUT_LO [1:0]	Input	

Signal	Direction	Description
RWDS_OE [1:0]	Input	Read/write data strobe output enable port.
RWDS_IN_HI [1:0]	Output	Read/write data strobe output ports for latency indication, also center-aligned reference strobe for read data. Registered in normal mode/resync mode of DDIO.
RWDS_IN_LO [1:0]	Output	
DQ_OUT_HI [15:0]	Input	DQ input ports for command, address and data. Registered in normal mode of DDIO.
DQ_OUT_LO [15:0]	Input	
DQ_OE [15:0]	Input	DQ output enable port.
DQ_IN_HI [15:0]	Output	DQ output ports for data.
DQ_IN_LO [15:0]	Output	

Using the HyperRAM Interface

To use the HyperRAM block, you add the block, choose the resource and specify the instance and pin names. Then, add a HyperRAM Controller IP core instance to your project to connect it to the HyperRAM. A PLL generates the control signals.

Figure 37: Example System with HyperRAM Block



Note: You can generate an example design with the IP Manager. Open the wizard for the HyperRAM Controller IP core and select **Deliverables** > **Example Design (Ti60F100_pll_cal)** and generate. Refer to the HyperRAM Controller IP Core User Guide for a description of this example.

Table 52: HyperRAM Interface Designer Settings

Option	Values	Notes
Instance Name	User defined	
HyperRAM Resource	HYPER_RAM0	
Active-Low HyperRAM Reset Drive Strength (mA)	4, 8, 12, 16	Default: 4. Choose the drive strength current in mA.
Active-Low HyperRAM Chip Select Drive Strength (mA)	4, 8, 12, 16	Default: 4. Choose the drive strength current in mA.
HyperRAM Clock Drive Strength (mA)	4, 8, 12, 16	Default: 4. Choose the drive strength current in mA.
Read/Write Data Strobe Drive Strength (mA)	4, 8, 12, 16	Default: 4. Choose the drive strength current in mA.

Option	Values	Notes
DQ [15:0] Bus Strength (mA)	4, 8, 12, 16	Default: 4. Choose the drive strength current in mA.
HyperRAM Controller Clock Pin Name	User defined	Must come from PLL output
Calibration Clock Pin Name	User defined	Must come from PLL output
90 Degree Phase-Shifted Clock Pin Name	User defined	Must come from PLL output
Active-Low HyperRAM Reset Pin Name	User defined	
Active-Low HyperRAM Chip Select Pin Name	User defined	
Differential Clock Pin Name (P HI)	User defined	
Differential Clock Pin Name (P LO)	User defined	
Differential Clock Pin Name (N HI)	User defined	
Differential Clock Pin Name (N LO)	User defined	
Read/Write Data Strobe Output [1:0] Bus Name (HI)	User defined	
Read/Write Data Strobe Output [1:0] Bus Name (LO)	User defined	
Read/Write Data Strobe Output Enable [1:0] Bus Name	User defined	
Read/Write Data Strobe Input [1:0] Bus Name (HI)	User defined	
Read/Write Data Strobe Input [1:0] Bus Name (LO)	User defined	
DQ Output [15:0] Bus Name (HI)	User defined	
DQ Output [15:0] Bus Name (LO)	User defined	
DQ Output Enable [15:0] Bus Name	User defined	
DQ Input [15:0] Bus Name (HI)	User defined	
DQ Input [15:0] Bus Name (LO)	User defined	

Design Check: HyperRAM Messages

When you check your design, the Interface Designer applies design rules to your HyperRAM settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

hyper_ram_rule_cali_clk_dyn_phase_shift (error)

Message	Dynamic Phase Shift should be enabled in the PLL (<pll_inst.name>:CLKOUT<pll_clk_idx>) that drives the HyperRAM's Calibration Clock
To fix	Enable the Dynamic Phase Shift in the PLL and the PLL output clock that drives the Calibration Clock.

hyper_ram_rule_cali_clk_phase_shift_coverage (warning)

Message	The recommended phase shift step for the Calibration Clock is 45 degrees. Current: <phase_shift_step> degrees.
To fix	Update the PLL settings so the step for the dynamic phase shift in Calibration Clock has 45 degree phase coverage. The formula to calculate a single phase shift step coverage is given by: Single phase step coverage = (0.5 x Post Divider (O) x Final Clock Out) / VCO Frequency x 360

hyper_ram_rule_cali_clk_pll (error)

Message	Calibration Clock is not driven by PLL
To fix	Update the PLL settings such that the Calibration Clock is driven by a PLL.

Message	Calibration Clock and Controller Clock must have identical frequency
To fix	Update the PLL settings such that the Controller Clock and Calibration Clock have the same frequency.

hyper_ram_rule_clk_freq (error)

Message	Clock signals: <Clock Names> driving HyperRAM must have frequencies less than or equal to 250 MHz.
To fix	Update the PLL settings to reduce the frequency of the clocks driving the HyperRAM.

hyper_ram_rule_drive_strength (error)

Message	The following drive strength are invalid: *<list of invalid drive strength params>*
To fix	Choose or specify a valid drive strength. See **Table 52: HyperRAM Interface Designer Settings** on page 113.

hyper_ram_rule_empty_pins (error)

Message	Empty pin names found: <list>
To fix	Specify the missing pin names in the list.

hyper_ram_rule_instance_count (error)

Message	There can only be one HyperRAM instance
To fix	You get this error when you create more than one HyperRAM instance.

hyper_ram_rule_invalid_pins (error)

Message	Invalid pin names found
To fix	The pin name you entered has illegal characters. Rename the pin.

hyper_ram_rule_pll_clk (warning)

Message	Phase shift relationship between Controller Clock and Phase-Shifted Clock is not guaranteed when they don't come from PLL
To fix	You get this error when either Controller Clock or Phase-Shifted Clock is not coming from PLL. Update the PLL settings such that both Controller Clock and Phase-Shifted Clock comes from PLL

hyper_ram_rule_pll_clk (error)

Message	Controlled Clock and Phase-Shifted Clock cannot be the same
To fix	Specify different names for the Controller Clock and Phase-Shifted Clock.

Message	Controller Clock and Phase-Shifted Clock should connect to the same PLL instance
To fix	Connect the Controller Clock and Phase-Shifted Clock to the same PLL instance.

Message	Invalid phase shift difference: <phase_diff> between CLK90: <shifted_clk_degree> and CLK: <ctrl_clk_degree>) (Expected 90 degrees)
To fix	Update the PLL settings such that the phase shift difference between Controller Clock and Phase-Shifted Clock is 90 degree.

Message	Controller Clock and Phase-Shifted Clock must have identical frequency
To fix	Update the PLL settings such that the Controller Clock and Phase Shifted Clock have the same frequency.

hyper_ram_rule_resource (error)

Message	Resource name is empty
To fix	The HyperRAM resource name.

Message	Resource is not a valid HyperRAM device instance
To fix	Choose the HyperRAM resource.

Chapter 8

JTAG User TAP Interface

Contents:

Titanium FPGAs have dedicated JTAG pins to for configuration and boundary scan testing.

JTAG Mode

The JTAG serial configuration mode is popular for prototyping and board testing. The four-pin JTAG boundary-scan interface is commonly available on board testers and debugging hardware.

Table 53: Supported Topaz and Titanium JTAG Instructions

Instruction	Binary Code [4:0]	Description
BYPASS	11111	Enables BYPASS.
DEVICE_STATUS	01100	Lets you read the device configuration status.
EFUSE_PREWRITE	11000	Loads user data for fuse operations.
EFUSE_USER_WRITE	11010	Blows fuses as defined in EFUSE_PREWRITE.
EFUSE_WRITE_STATUS	11011	Returns status of EFUSE_USER_WRITE operation.
ENTERUSER	00111	Changes the FPGA into user mode.
EXTEST	00000	Enables the boundary-scan EXTEST operation.
IDCODE	00011	Enables shifting out the IDCODE.
INTEST	00001	Enables the boundary-scan INTEST operation.
JTAG_USER1	01000	Connects the JTAG User TAP 1.
JTAG_USER2	01001	Connects the JTAG User TAP 2.
JTAG_USER3	01010	Connects the JTAG User TAP 3.
JTAG_USER4	01011	Connects the JTAG User TAP 4.
PROGRAM	00100	JTAG configuration.
SAMPLE/PRELOAD	00010	Enables the boundary-scan SAMPLE/PRELOAD operation.
USERCODE	01101	Use this instruction to program a 32-bit signature into the FPGA during programming.

Note: For detailed information about using JTAG for configuration, refer to AN 033: Configuring Titanium FPGAs.

Using the JTAG User TAP Block

Add the JTAG User TAP block to your interface if you want to use the FPGA JTAG pins to communicate with the design running in the core.

You specify the instruction to use with the **JTAG Resource** setting. Titanium FPGAs have four JTAG User TAP blocks. To use more than one, add JTAG User TAP blocks to your interface design, one for each resource.

Table 54: JTAG User TAP Signals

Signal	Direction	Description
<instance>_TDI	Input	JTAG test data in pin.
<instance>_TCK	Input	JTAG test clock pin.
<instance>_TMS	Input	JTAG mode select pin.
<instance>_SEL	Input	User instructive active pin.
<instance>_DRCK	Input	Gated test clock.
<instance>_RESET	Input	Reset.
<instance>_RUNTEST	Input	Run test pin.
<instance>_CAPTURE	Input	Capture pin.
<instance>_SHIFT	Input	Shift pin.
<instance>_UPDATE	Input	Update pin.
<instance>_TDO	Output	JTAG test data out pin.

Design Check: JTAG User Tap Messages

When you check your design, the Interface Designer applies design rules to your JTAG User Tap settings. The following tables show some of the error and warnng messages you may encounter and explains how to fix them.

jtag_rule_clock (warning)

Message	Clock pin name is empty
To fix	Enter a valid clock pin name.

jtag_rule_resource (error)

Message	Resource name is empty Resource is not a valid JTAG device instance
To fix	Specify a valid JTAG resource.

Chapter 9

MIPI RX/TX Lane Interface

Contents:

Each HSIO block can use a pair of I/O pins as a MIPI RX or TX data lane or clock lane.

HSIO Configured as MIPI Lane

You can configure the HSIO block as a MIPI RX or TX lane. The block supports bidirectional data lane, unidirectional data lane, and unidirectional clock lane which can run at speeds up to 1.5 Gbps. The MIPI lane operates in high-speed (HS) and low-power (LP) modes. In HS mode, the HSIO block transmits or receives data with x8 serializer/deserializer. In LP mode, it transmits or receives data without deserializer/serializer.

The MIPI lane block does not include the MIPI D-PHY core logic. A full MIPI D-PHY solution requires:

- Multiple MIPI RX or TX lanes (at least a clock lane and a data lane)
- Soft MIPI D-PHY IP core programmed into the FPGA fabric

The MIPI D-PHY standard is a point-to-point protocol with one endpoint (TX) responsible for initiating and controlling communication. Often, the standard is unidirectional, but when implementing the MIPI DSI protocol, you can use one TX data lane for LP bidirectional communication.

The protocol is source synchronous with one clock lane and 1, 2, 4, or 8 data lanes. The number of lanes available depends on which package you are using. A dedicated HSIO block is assigned on the RX interface as a clock lane while the clock lane for TX interface can use any of the HSIO block in the group.

MIPI RX Lane

In RX mode, the HS (fast) clock comes in on the MIPI clock lane and is divided down to generate the slow clock. The fast and slow clocks are then passed to neighboring HSIO blocks to be used for the MIPI data lanes.

The data lane fast and slow clocks must be driven by a clock lane in the same MIPI group (dedicated buses drive from the clock lane to the neighboring data lanes).

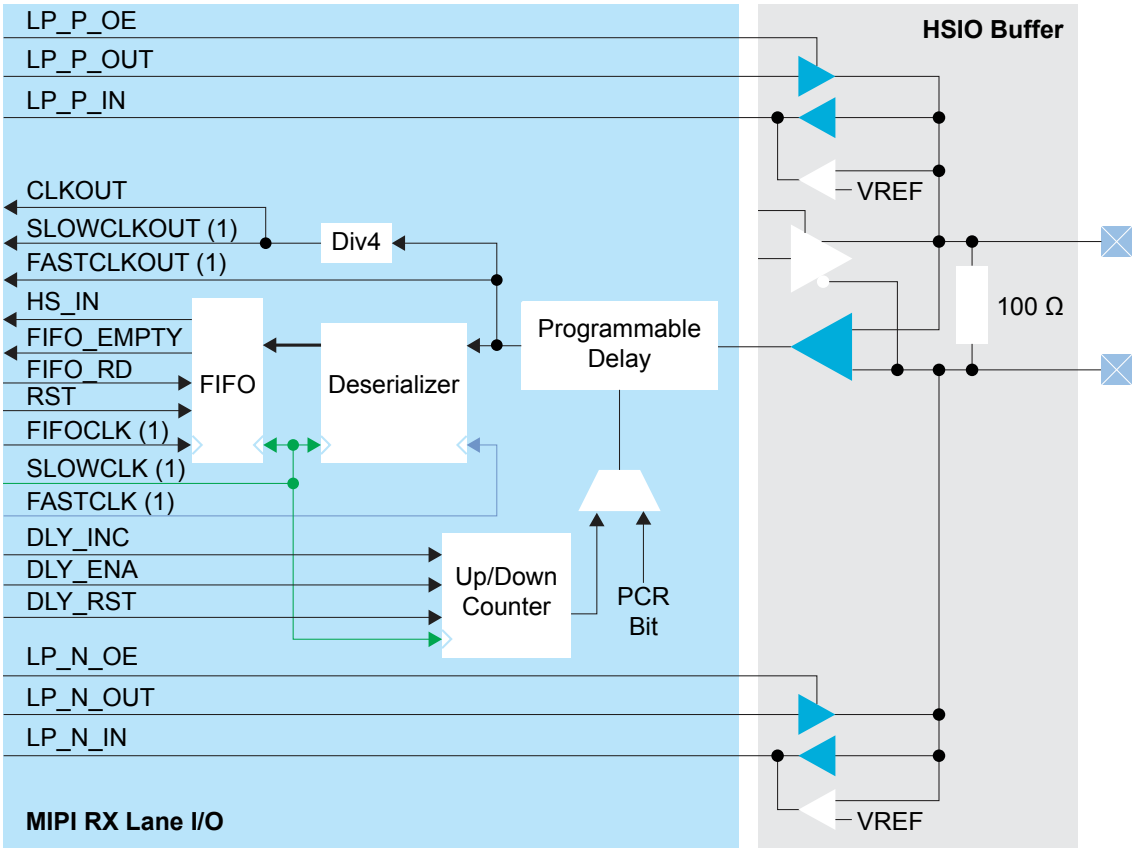
The MIPI RX function is defined as:

Table 55: MIPI RX Function

MIPI RX Function	Description
RX_DATA_*xy_zz*	MIPI RX Data Lane. You can use any data lanes within the same group to form multiple lanes of MIPI RX channel. *x* = P or N *y* = 0 to 7 data lanes (Up to 8 data lanes per channel) **Ti35, Ti60** *zz* = I0 to I11 MIPI RX channel (Up to 12 MIPI RX channels) **Ti85, Ti135, Ti90, Ti120, Ti165, Ti180, Ti240, Ti375** *zz* = I0 to I17 MIPI RX channel (Up to 18 MIPI RX channels)
RX_CLK_*x_zz*	MIPI RX Clock Lane. One clock lane is required for each MIPI RX channel. *x* = P or N **Ti35, Ti60** *zz* = I0 to I11 MIPI RX channel (Up to 12 MIPI RX channels) **Ti85, Ti135, Ti90, Ti120, Ti165, Ti180, Ti240, Ti375** *zz* = I0 to I17 MIPI RX channel (Up to 18 MIPI RX channels)

Learn more: Refer to the pinout file for your FPGA for more information about the MIPI RX function for each HSIO and for which pins are in the same MIPI group.

Figure 38: MIPI RX Lane Block Diagram



1. These signals are in the primitive, but the software automatically connects them for you.

Table 56: MIPI RX Lane Signals

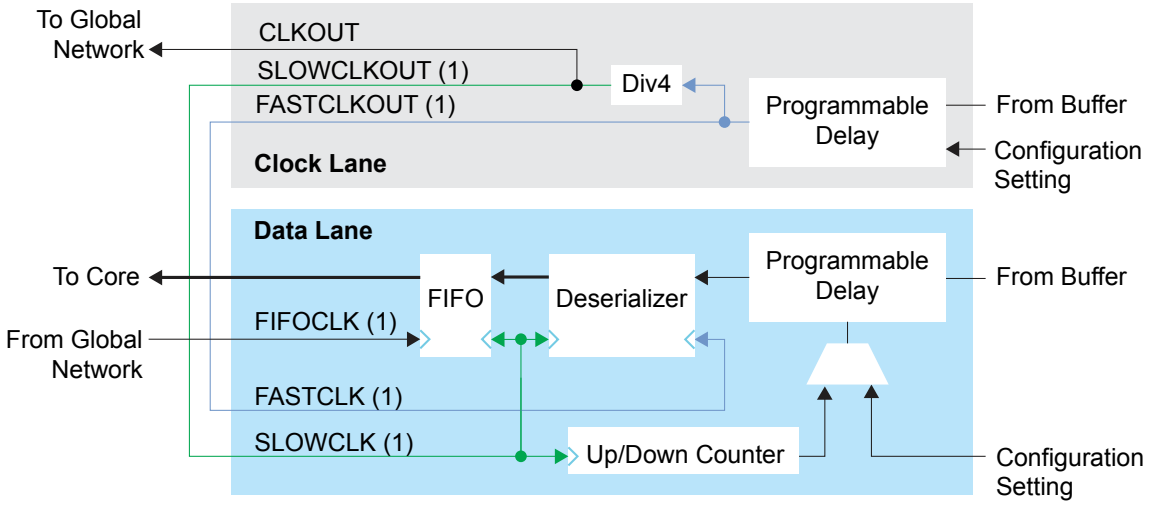
Interface to MIPI soft CSI/DSI controller with D-PHY in FPGA Fabric

Signal	Direction	Clock Domain	Description
LP_P_OE	Input	–	(Optional) LP output enable signal for P pad.
LP_P_OUT	Input	–	(Optional) LP output data from the core for the P pad. Used if the data lane is reversible.
LP_P_IN	Output	–	LP input data from the P pad.
CLKOUT	Output	–	Divided down parallel (slow) clock from the pads that can drive the core clock tree. Used to drive the core logic implementing the rest of the D-PHY protocol. It should also connect to the FIFOCLK of the data lanes.
SLOWCLKOUT[12]	Output	–	Divided down parallel (slow) clock from the pads. Can only drive RX DATA lanes.
FASTCLKOUT[12]	Output	–	Serial (fast) clock from the pads. Can only drive RX DATA lanes.
HS_IN[7:0]	Output	SLOWCLK	High-speed parallel data input.
FIFO_EMPTY	Output	FIFOCLK	(Optional) When the FIFO is enabled, this signal indicates that the FIFO is empty.
FIFO_RD	Input	FIFOCLK	(Optional) Enables FIFO to read.
RST	Input	FIFOCLK SLOWCLK	(Optional) Asynchronous. Resets the FIFO and serializer. If the FIFO is enabled, it is relative to FIFOCLK; otherwise it is relative to SLOWCLK.
FIFOCLK[12]	Input	–	(Optional) Core clock to read from the FIFO.
SLOWCLK[12]	Input	–	Parallel (slow) clock.
FASTCLK[12]	Input	–	Serial (fast) clock.
DLY_INC	Input	SLOWCLK	(Optional) Dynamic delay control. When DLY_ENA is 1, 1: Increments 0: Decrements
DLY_ENA	Input	SLOWCLK	(Optional) Enable the dynamic delay control.
DLY_RST	Input	SLOWCLK	(Optional) Reset the delay counter.
LP_N_OE	Input	–	(Optional) LP output enable signal for N pad.
LP_N_OUT	Input	–	(Optional) LP output data from the core for the N pad. Used if the data lane is reversible.
LP_N_IN	Output	–	LP input data from the N pad.
HS_ENA	Input	–	Dynamically enable the differential input buffer when in high-speed mode.
HS_TERM	Input	–	Dynamically enables input termination high-speed mode.

[12] These signals are in the primitive, but the software automatically connects them for you.

The clock lane generates the fast clock and slow clock for the RX data lanes within the interface group. It also generates a clock which is divided by 4 that feeds the global network. The following figure shows the clock connections between the clock and data lanes.

Figure 39: Connections for Clock and RX Data Lane in the Same MIPI RX Channel



1. The software automatically connects this signal for you.

MIPI TX Lane

In TX mode, a PLL generates the parallel and serial clocks and passes them to the clock and data lanes.

Figure 40: MIPI TX Lane Block Diagram

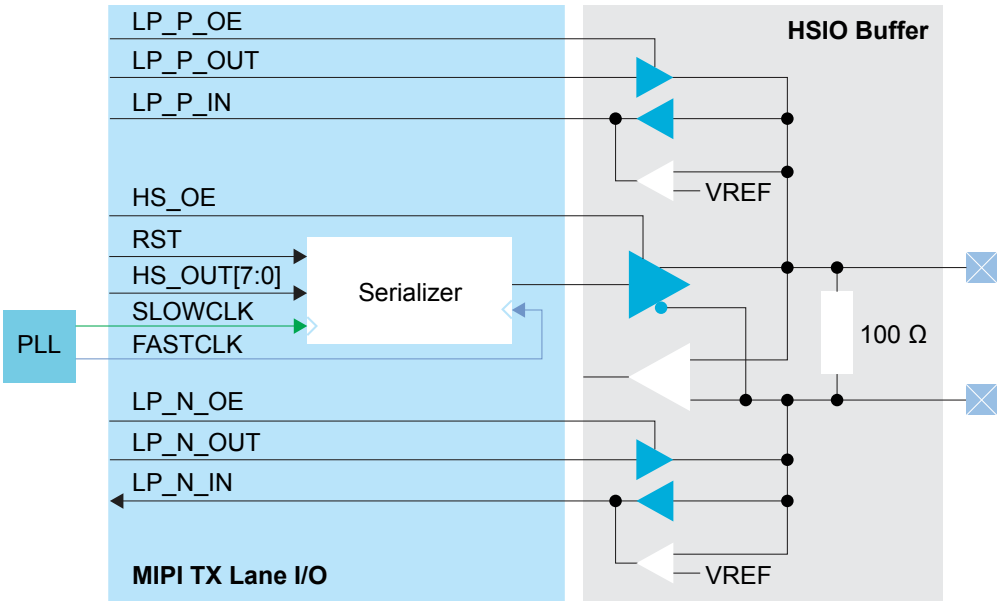


Table 57: MIPI TX Lane Signals

Interface to MIPI soft CSI/DSI controller with D-PHY in FPGA fabric

Signal	Direction	Clock Domain	Description
LP_P_OE	Input	–	LP output enable signal for P pad.
LP_P_OUT	Input	–	LP output data from the core for the P pad.
LP_P_IN	Output	–	(Optional) LP input data from the P pad. Used if data lane is reversible.
HS_OE	Input	–	High-speed output enable signal.
RST	Input	SLOWCLK	(Optional) Resets the serializer.
HS_OUT[7:0]	Input	SLOWCLK	High-speed output data from the core. Always 8-bits wide.
SLOWCLK	Input	–	Parallel (slow) clock.
FASTCLK	Input	–	Serial (fast) clock.
LP_N_OE	Input	–	LP output enable signal for N pad.
LP_N_OUT	Input	–	LP output data from the core for the N pad.
LP_N_IN	Output	–	(Optional) LP input data from the N pad. Used if data lane is reversible.

MIPI Lane Pads

Table 58: MIPI Lane Pads

Signal	Direction	Description
P	Output	Differential pad P.
N	Output	Differential pad N.

MIPI Groups by Package

You can use multiple HSIO as MIPI D-PHY lanes to build complete MIPI interfaces with one clock lane and up to 8 data lanes.

- For MIPI TX interfaces, you can use any lane anywhere on the FPGA.
- For MIPI RX interfaces, the number of data lanes is restricted by the number of lanes in the MIPI group. These groups vary depending on the package.

The Resource Assigner shows the MIPI RX group in the Block Summary's **Feature** field.

Learn more: To view the MIPI RX groups for each package graphically, go to the Titanium Packaging User Guide .

Using the MIPI TX Lane or MIPI RX Lane Block

The following tables show how to implement a MIPI TX Lane or MIPI RX Lane block. Later sections explain how to build a complete interface.

MIPI TX Lane Block

Table 59: MIPI TX Lane Block

Option	Choices	Description
Instance Name	User defined	Type the instance name and press enter.
MIPI Lane Resource	Resource list	Choose a resource
Mode	data lane, clock lane	Choose whether the block is a clock lane or data lane.
Enable LP Reverse Communication	On or off	When on, specify the low-power N and P pins.
Pin names (various)	User defined	Specify the interface bus and pin names.
Serial Clock Pin Name	User defined	Name you are using for FASTCLK_D or FASTCLK_C.
Parallel Clock Pin Name	User defined	Name you are using for SLOWCLK.
Static Delay Mode Setting	0 - 63	Choose the amount of static delay, each step adds approximately 25 ps of delay.

MIPI RX Lane Block

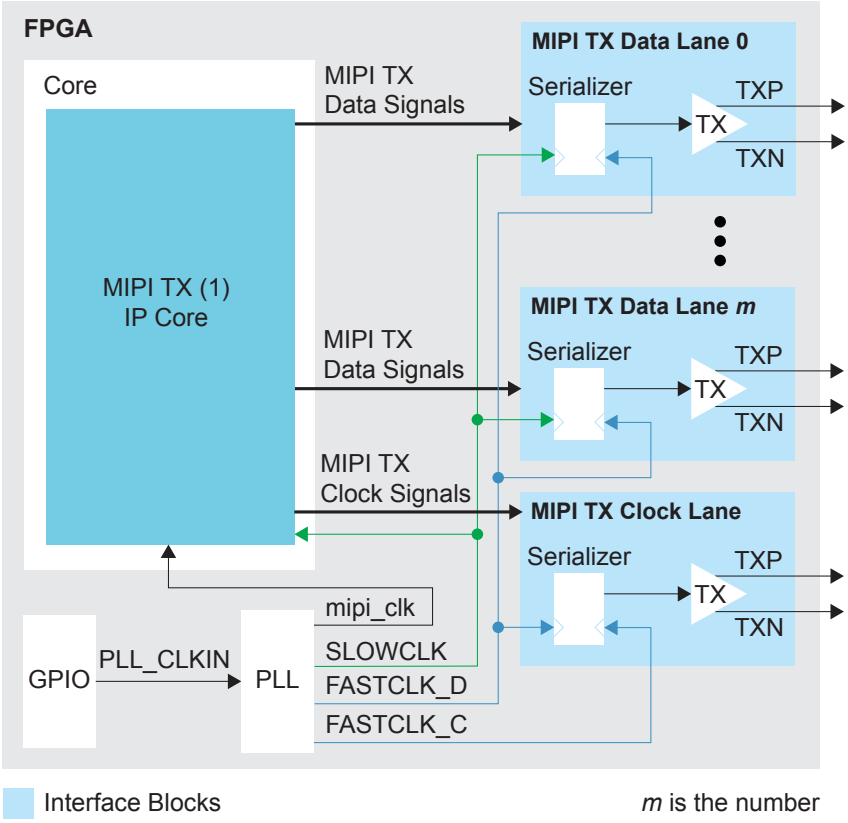
Table 60: MIPI RX Lane Block

Option	Choices	Description
Instance Name	User defined	Type the instance name and press enter.
MIPI Lane Resource	Resource list	Choose a resource.
Mode	data lane, clock lane	Choose whether the block is a clock lane or data lane.
Connection Type	gclk, rclk	In **clock lane** mode, choose global clock (gclk) or regional clock (rclk).
Enable LP Reverse Communication	On or off	When on, specify the low-power N and P pins.
Enable Clock Crossing FIFO	On or off	When on, specify the FIFO read and empty pins.
Pin names (various)	User defined	Specify the interface bus and pin names.
Delay Mode	static	Integer from 0 - 63. Each step adds approximately 25 ps of delay.
	dynamic	Specify the pin names to control the dynamic delay.

Create a MIPI TX Interface

To build a complete MIPI TX interface you need to have at least one data lane and one clock lane. Unlike MIPI RX, they can be in *any* MIPI group. The following figure shows the blocks used for a complete MIPI TX interface.

Figure 41: MIPI TX Interface



1. Refer to the Efinity® Software User Guide for a listing of available MIPI-related IP cores.

Important: You need to use specific phase shifts for the `SLOWCLK`, `FASTCLK_C`, and `FASTCLK_D` output clocks from the PLL as shown in step 1 below.

1. Add a PLL block with the following settings:

Option	Description
Resource	You can use any PLL resource.
Reference Clock Mode	External or Core.
Feedback Mode	Core. CLKOUT 1 to 3 from the PLL can be used for the feedback as long as the frequency and phase shift can be generated.
Reference Clock Frequency	User defined.
Output Clocks	mipi_clk–Frequency defined in MIPI IP core, phase shift 0°[13]

[13] This PLL also generates the `mipi_clk`, which is used in the MIPI IP core. Refer to the user guide for the IP core for details.

Option	Description
	SLOWCLK–Frequency is 1/8 the PHY speed, phase shift 0°, enable feedback. FASTCLK_D–Frequency is the speed you are running the PHY, phase shift 90.00° FASTCLK_C–Frequency is the speed you are running the PHY, phase shift 180.00° For example, if the PHY is running at 1,000 Mbps, FASTCLK_D and FASTCLK_C will run at half that 500 MHz (because it transfers data on both clock edges), and SLOWCLK will run at 125 MHz.
Locked Pin	Turn on

2. Add a GPIO block with these settings to provide the reference clock input to the PLL:

Option	Description
Mode	Input
Pin Name	Any
Connection Type	pll_clkin
GPIO Resource	Assign the dedicated PLL_CLKIN pin that corresponds to the PLL you chose.

3. Add MIPI TX Lane block with these settings:

Option	Description
Mode	data lane
Parallel Clock Pin Name	Name you are using for SLOWCLK.
Serial Clock Pin Name	Name you are using for FASTCLK_D.

4. Repeat step 3 for each MIPI TX data lane you want to implement.
5. Add another MIPI TX Lane block for the clock lane:

Option	Description
Mode	clock lane
Parallel Clock Pin Name	Name you are using for FASTCLK_D.
Serial Clock Pin Name	Name you are using for FASTCLK_C.

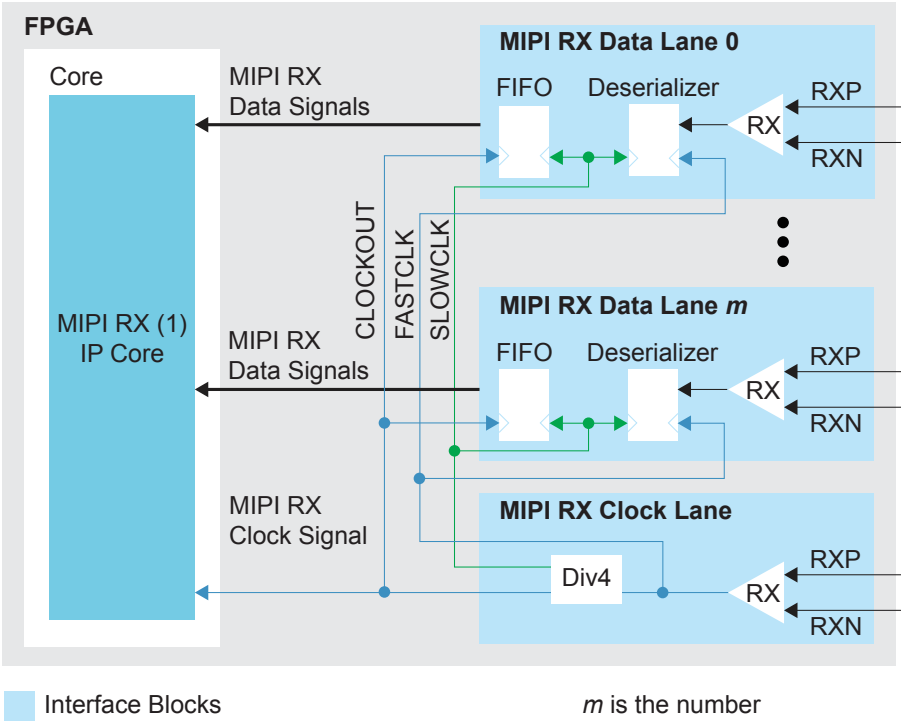
6. Implement the rest of the MIPI TX interface in RTL using a MIPI TX IP core (CSI-2, D-PHY, or DSI). Refer to the user guide for the IP core for instructions.

Create a MIPI RX Interface

To build a complete MIPI RX interface you need to have at least one data lane and one clock lane in the *same* MIPI group. The following figure shows the blocks used for a complete MIPI RX interface.

Tip: The Interface Designer Block Summary shows the MIPI group name in the **Features** property. You can also refer to MIPI Groups by Package on page 124.

Figure 42: MIPI RX Interface



1. Refer to the Efinity® Software User Guide for a listing of available MIPI-related IP cores.

1. Add MIPI RX Lane block for the data lane:

Option	Description
MIPI Lane Resource	Choose a resource in any MIPI group (all lanes should be in the same MIPI group).
Mode	data lane

2. Repeat step 1 for each MIPI RX data lane you want to implement.
3. Add another MIPI RX Lane block for the clock lane:

Option	Description
MIPI Lane Resource	Choose a resource in the same MIPI group as the data lane(s).
Mode	clock lane
Byte Clock (core) Pin Name	The name for the byte clock that feeds the core and automatically is used to clock the FIFO from the data lanes in this MIPI group.

4. Implement the rest of the MIPI RX interface in RTL using a MIPI RX IP core (CSI-2, D-PHY, or DSI). Refer to the user guide for the IP core for instructions.

Design Check: MIPI Lane Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error and warning messages you may encounter and explains how to fix them.

mipi_ln_rule_resource (error)

Message	Resource name is empty Resource <name> is not a valid MIPI LANE <Rx/Tx> device instance
To fix	Choose a valid resource.

mipi_ln_rule_group_clock (error)

Message	No clock lane was configured in the MIPI LANE group <group> No clock lane was configured in the same MIPI LANE group
To fix	The minimum requirement to create a MIPI interface is one clock lane and one data lane. For RX, they must also be in the same MIPI group. Add a clock lane.

mipi_ln_rule_group_data (error)

Message	Instance is not a resource associated to a MIPI LANE Rx group Instance is not a valid MIPI LANE data lane resource
To fix	In some packages there are not enough MIPI resources in a group to have both a clock and a data lane for RX interfaces. In those cases, you cannot use that pin as a MIPI lane. Choose another resource. See **MIPI Groups by Package** on page 124.

mipi_ln_rule_rx_clk_conn (error)

Message	Connection type <type> is not supported by the resource The resource does not support clock lane mode
To fix	Not all resources support both GCLK and RCLK connection types. Use the Resource Assigner to pick a different resource that supports GCLK and RCLK.

mipi_ln_rule_rx_distance (error)

Message	These HSIO GPIO must be placed at least 1 pair away from MIPI LANE <name> in order to avoid noise coupling from GPIO to MIPI LANE: <violated list>
To fix	When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO used as MIPI RX lanes in the same bank. This separation reduces noise.

mipi_ln_rule_rx_empty_pins (error)

Message	Empty pin names found: <list>
To fix	Specify the missing pin names in the list.

mipi_ln_rule_rx_param (error)

Message	Invalid parameters configuration: <features>
To fix	One of the parameters you set was incorrect. Review any other errors for details.

mipi_ln_rule_tx_clock (error)

Message	Serial and parallel clocks cannot be the same clock
To fix	You cannot use the same clock for both the serial (FASTCLK_C or FASTCLK_D) and parallel (SLOWCLK) clocks.

Message	Serial clock name is not a PLL output clock
To fix	Use a PLL output clock as the serial (FASTCLK_C or FASTCLK_D) clock.

Message	Parallel clock name is not a PLL output clock
To fix	Use a PLL output as the parallel (SLOWCLK) clock.

Message	Serial and parallel clocks are not from the same PLL instance
To fix	You need to use the same PLL to generate both clocks.

Message	Expected clocks phase shift in <data/clock> mode: Serial: <int> degree, Parallel: <int> degree
To fix	You need to set the phase shift for the lane in clock or data model. This phase shift varies depending on the serial clock frequency (less than 500 MHz or otherwise). If your PLL is running at 500 or lower, then you can use the old 45/135 rule and it will be oK. for anything higher than that, you need to use the new 90/180 rule.

Message	Expected clocks phase shift in <data/clock> mode: Valid Serial (S),Parallel (P) Combinations: (S:135 degree, P:0 degree) OR (S:180 degree, P:90 degree)
To fix	Clock lanes with a serial frequency greater than or equal to 500 MHz must use the specified phase shift settings.

Message	One of the clock frequencies is 0
To fix	The output clock frequency is invalid. FASTCLK_D and FASTCLK_C should be the same frequency as the PHY. SLOWCLK should be 1/8 the PHY frequency. For example, if the PHY is running at 800 MHz, FASTCLK_D and FASTCLK_C should be 800 MHz and SLOWCLK should be 100 MHz.

Message	Serial clock frequency has to be <n> times faster than parallel clock
To fix	FASTCLK_D and FASTCLK_C should be the same frequency as the PHY. SLOWCLK should be 1/8 the PHY frequency. For example, if the PHY is running at 800 MHz, FASTCLK_D and FASTCLK_C should be 800 MHz and SLOWCLK should be 100 MHz.

mipi_ln_rule_tx_clock_region (error)

Message	Serial and Parallel clocks generated by PLL have to be driven to the same clock network. <Serial | Parallel> clock <name> was generated by PLL output clock 4 that connects to regional clock network
To fix	In Ti35, Ti60 FPGAs, the PLL's output clock 4 can only drive the regional clock network. You should use the other clock outputs for the serial and parallel clocks.

lvds_rule_tx_distance (error)

Message	These HSIO GPIO must be placed at least 1 pair away from MIPI LANE <name> in order to avoid noise coupling from GPIO to MIPI LANE: <violated list>
To fix	When using HSIO pins as GPIO, make sure to leave at least 1 pair of unassigned HSIO pins between any GPIO and HSIO used as MIPI TX lanes in the same bank. This separation reduces noise.

mipi_ln_rule_tx_empty_pins (error)

Message	Empty pin names found: <list>
To fix	Specify the missing pin names in the list.

mipi_ln_rule_tx_param (error)

Message	Invalid parameters configuration: <features>
To fix	One of the parameters you set was incorrect. Review any other errors for details.

mipi_ln_rule_usage (error)

Message	Resource <res name> was assigned multiple times
To fix	You get this error if you choose the same resource for more than one block type (LVDS, MIPI DPHY, or GPIO).

Chapter 10

MIPI D-PHY Interface

Contents:

Important: Ti85, Ti135, Ti165, Ti240, Ti375: All information is preliminary and pending definition.

In addition to the HSIO, which you can configure as MIPI RX or TX lanes, Titanium FPGAs have hardened MIPI D-PHY blocks, each with 4 data lanes and 1 clock lane. The MIPI D-PHY RX and MIPI D-PHY TX can operate independently with dedicated I/O banks.

You can use the hardened MIPI D-PHY blocks along with the HSIO configured as MIPI D-PHY lanes to create systems that aggregate data from many cameras or sensors.

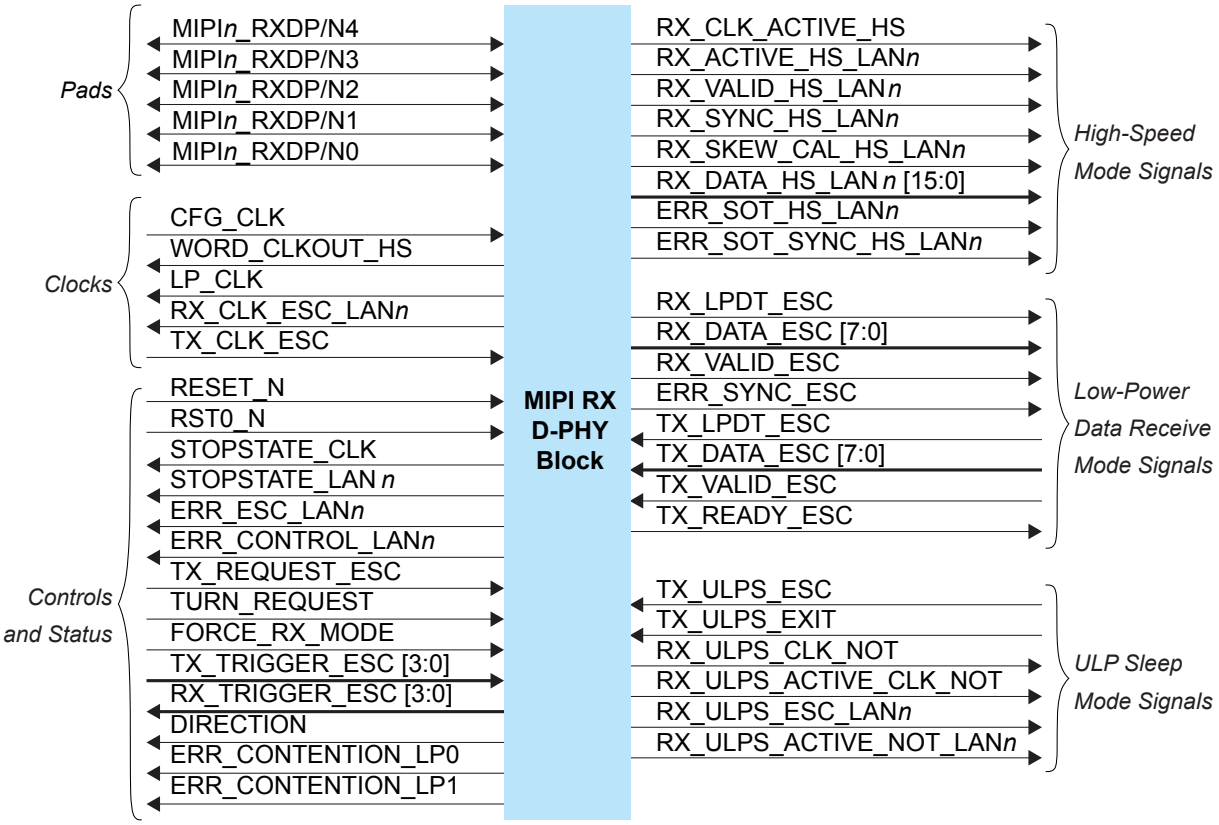
The MIPI TX/RX interface supports the MIPI D-PHY specification v1.1. It has the following features:

- Programmable data lane configuration supporting up to 4 lanes
- High-speed mode supports up to 2.5 Gbps data rates per lane
- Operates in continuous and non-continuous clock modes
- Supports Ultra-Low Power State (ULPS)

MIPI RX D-PHY

The MIPI RX D-PHY is a receiver interface designed to receive data and the control information of MIPI CSI, DSI, or other associated protocols. The MIPI RX D-PHY comprises of one clock lane and up to four data lanes for a single-channel configuration. The MIPI RX D-PHY also interfaces with MIPI-associated protocol controllers via a standard MIPI D-PHY PHY Protocol Interface (PPI) that supports the 8- or 16-bit high-speed receiving data bus.

Figure 43: MIPI RX D-PHY x4 Block Diagram



The status signals provide optional status and error information about the MIPI RX D-PHY interface operation.

Figure 44: MIPI RX D-PHY Interface Block Diagram

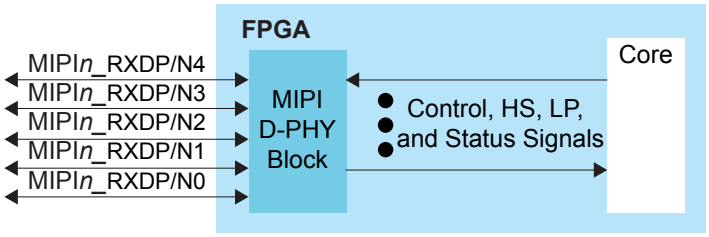


Table 61: MIPI RX D-PHY Clocks Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
CFG_CLK	Input	N/A	Configuration Clock (used for time counter and EQ calibration). The clock must be between 80 MHz to 120 MHz.
WORD_CLKOUT_HS	Output	N/A	HS Receive Byte/Word clock.
LP_CLK	Output	N/A	Low Power State clock.
RX_CLK_ESC_LANn	Output	N/A	Escape Mode Receive clock.
TX_CLK_ESC	Input	N/A	Escape Mode Transmit clock. The clock must be lower than 20 MHz.

Table 62: MIPI RX D-PHY Control and Status Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
RESET_N	Input	N/A	Reset. Disables PHY and reset the digital logic.
RST0_N	Input	N/A	Asynchronous FIFO reset and synchronous out of reset.
STOPSTATE_CLK	Output	N/A	Lane in Stop State.
STOPSTATE_LAN*n*	Output	N/A	Data Lane in Stop State (Lane N).
ERR_ESC_LAN*n*	Output	N/A	Lane *n* Escape Command Error.
ERR_CONTROL_LAN*n*	Output	N/A	Lane *n* Has Line State Error.
TX_REQUEST_ESC	Input	TX_CLK_ESC	Lane 0 Request TX Escape Mode.
TURN_REQUEST	Input	TX_CLK_ESC	Lane 0 Request Turnaround.
FORCE_RX_MODE	Input	N/A	Lane 0 Force Lane into Receive Mode/Wait for Stop State.
TX_TRIGGER_ESC [3:0]	Input	TX_CLK_ESC	Lane 0 Send a Trigger Event.
RX_TRIGGER_ESC [3:0]	Output	RX_CLK_ESC_LAN0	Lane 0 Received a Trigger Event.
DIRECTION	Output	N/A	Lane 0 Transmit/Receive Direction (0 = TX, 1 = RX).
ERR_CONTENTION_LP0	Output	N/A	Lane 0 Contention Error when driving 0.
ERR_CONTENTION_LP1	Output	N/A	Lane 0 Contention Error when driving 1.

Table 63: MIPI RX D-PHY High-Speed Mode Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
RX_CLK_ACTIVE_HS	Output	N/A	HS Clock Lane Active.
RX_ACTIVE_HS_LAN*n*	Output	WORD_CLKOUT_HS	HS Reception Active.
RX_VALID_HS_LAN*n*	Output	WORD_CLKOUT_HS	HS Data Receive Valid.
RX_SYNC_HS_LAN*n*	Output	WORD_CLKOUT_HS	HS Reveiver Sync. Observed.
RX_SKEW_CAL_HS_LAN*n*	Output	WORD_CLKOUT_HS	HS Reveiver DeSkew Burst Received.
RX_DATA_HS_LAN*n* [15:0]	Output	WORD_CLKOUT_HS	HS Receive Data.
ERR_SOT_HS_LAN*n*	Output	WORD_CLKOUT_HS	State-of-Transmission (SOT) Error.
ERR_SOT_SYNC_HS_LAN*n*	Output	WORD_CLKOUT_HS	SOT Sync. Error.

Table 64: MIPI RX D-PHY Low-Power Data Receive Mode Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
RX_LPDT_ESC	Output	RX_CLK_ESC_LAN0	Lane 0 enter LPDT RX Mode.
RX_DATA_ESC [7:0]	Output	RX_CLK_ESC_LAN0	Lane 0 LPDT RX Data.
RX_VALID_ESC	Output	RX_CLK_ESC_LAN0	Lane 0 LPDT RX Data Valid.
ERR_SYNC_ESC	Output	N/A	Lane 0 LPDT RX Data Sync. Error.
TX_LPDT_ESC	Input	TX_CLK_ESC	Lane 0 Enter LPDT TX Mode.
TX_DATA_ESC [7:0]	Input	TX_CLK_ESC	Lane 0 LPDT TX Data.
TX_VALID_ESC	Input	TX_CLK_ESC	Lane 0 LPDT TX Data Valid.
TX_READY_ESC	Output	TX_CLK_ESC	Lane 0 LDPT TX Data Ready.

Table 65: MIPI RX D-PHY ULP Sleep Mode Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
TX_ULPS_ESC	Input	TX_CLK_ESC	Lane 0 Enter ULPS Mode.
TX_ULPS_EXIT	Input	TX_CLK_ESC	Lane 0 Exit ULPS Mode.
RX_ULPS_CLK_NOT	Output	N/A	CLK0 Enter ULPS Mode.
RX_ULPS_ACTIVE_CLK_NOT	Output	N/A	CLK0 is in ULPS (Active Low).
RX_ULPS_ESC_LAN*n*	Output	RX_CLK_ESC_LAN*n*	Lane *n* Enter ULPS Mode.
RX_ULPS_ACTIVE_NOT_LAN*n*	Output	N/A	Lane *n* is in ULPS (Active Low).

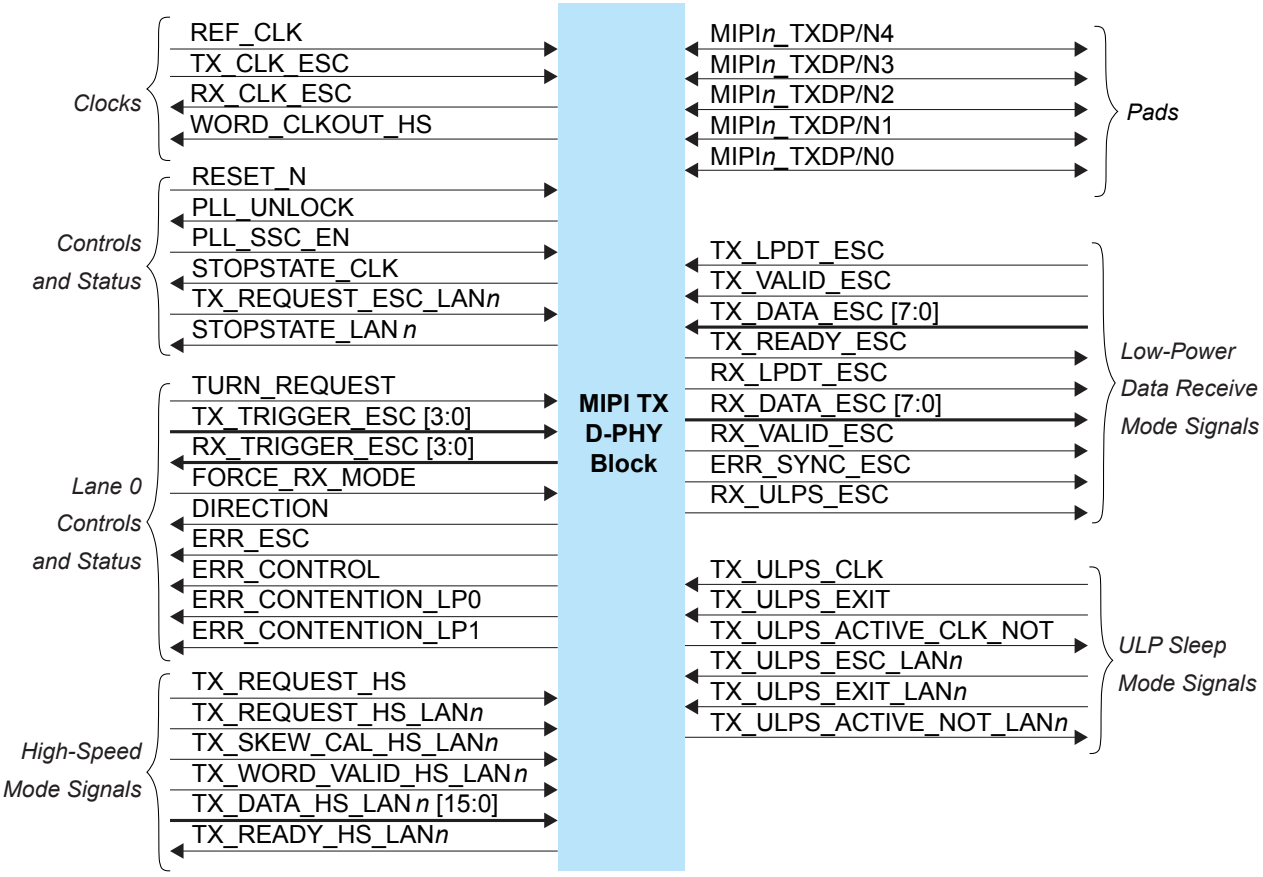
Table 66: MIPI RX D-PHY Pads

Pad	Direction	Description
MIPI*n*_RXDP[4:0]	Bidirectional	MIPI transceiver P pads.
MIPI*n*_RXDN[4:0]	Bidirectional	MIPI transceiver N pads.

MIPI TX D-PHY

The MIPI TX D-PHY is a transmitter interface designed to transmit data and the control information of MIPI CSI, DSI, or other associated protocols. The MIPI TX D-PHY comprises of one clock lane and up to four data lanes for a single-channel configuration. The MIPI TX D-PHY also interfaces with MIPI-associated protocol controllers via a standard MIPI D-PHY PPI that supports the 8- or 16-bit high-speed receiving data bus.

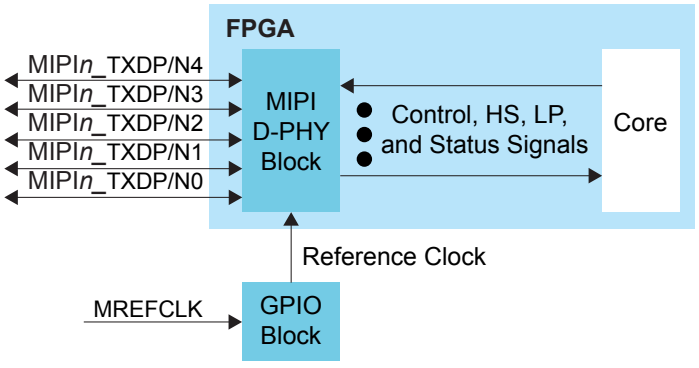
Figure 45: MIPI TX D-PHY x4 Block Diagram



The MIPI TX D-PHY block requires an escape clock (`TX_CLK_ESC`) for use when the MIPI interface is in escape (low-power) mode, which runs up to 20 MHz.

Note: Efinix recommends that you set the escape clock frequency as close to 20 MHz as possible.

Figure 46: MIPI TX D-PHY Interface Block Diagram



Note: GPIO block is the default reference clock source. However, the PLL and core clock out can also be set as the reference clock source.

Table 67: MIPI TX D-PHY Clocks Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
REF_CLK	Input	N/A	Reference Clock. The clock must be between 12 MHz to 52 MHz.
TX_CLK_ESC	Input	N/A	Escape Mode Transmit Clock, used to generate escape sequence. The clock must be less than 20 MHz.
RX_CLK_ESC	Output	N/A	Escape Mode Receive Clock (lane 0 only)
WORD_CLKOUT_HS	Output	N/A	HS Transmit Byte/Word Clock. This signal must be 1/8 of the bit-rate in normal 8-bit HS-PPI D-PHY mode, or 1/16 of the bit-rate in 16-bit PHY mode.

Table 68: MIPI TX D-PHY Control and Status Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
RESET_N	Input	N/A	Reset. Disables PHY and reset the digital logic.
PLL_UNLOCK	Output	N/A	PLL is in unlock state.
PLL_SSC_EN	Input	N/A	(Optional) PLL SSC Enable: Always enable: 1 Disable: 0 Driven by active signal for dynamic enable
STOPSTATE_CLK	Output	N/A	Clock Lane in Stop State (Clk 0).
TX_REQUEST_ESC_LAN*n*	Input	TX_CLK_ESC	Escape Mode Transmit Request (Lane N).
STOPSTATE_LAN*n*	Output	N/A	Data Lane in Stop State (Lane N).

Table 69: MIPI TX D-PHY Lane 0 Control and Status Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
TURN_REQUEST	Input	TX_CLK_ESC	Lane 0 Turnaround Request.
TX_TRIGGER_ESC [3:0]	Input	TX_CLK_ESC	Lane 0 Send an Escape Mode Trigger Event.
RX_TRIGGER_ESC [3:0]	Output	RX_CLK_ESC	Lane 0 Received an Escape Mode Trigger Event.
FORCE_RX_MODE	Input	N/A	Lane 0 Force into Receive Mode/Wait for Stop.
DIRECTION	Output	N/A	Lane 0 Transmit/Receive Direction: 0: TX, 1: RX
ERR_ESC	Output	N/A	Lane 0 Escape Command Error.
ERR_CONTROL	Output	N/A	Lane 0 Line State Error.
ERR_CONTENTION_LP0	Output	N/A	Lane 0 Line Contention Detected (when driving 0).
ERR_CONTENTION_LP1	Output	N/A	Lane 0 Line Contention Detected (when driving 1).

Table 70: MIPI TX D-PHY High Speed Mode Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
TX_REQUEST_HS	Input	WORD_CLKOUT_HS	HS Clock Request (Clk 0).
TX_REQUEST_HS_LAN*n*	Input	WORD_CLKOUT_HS	HS Transmit Request and Data Valid (Lane 0-3).
TX_SKEW_CAL_HS_LAN*n*	Input	WORD_CLKOUT_HS	HS Skew Calibration (Lane N).
TX_WORD_VALID_HS_LAN*n*	Input	WORD_CLKOUT_HS	HS High Byte Valid (Lane N) for 16-bit mode.
TX_DATA_HS_LAN*n* [15:0]	Input	WORD_CLKOUT_HS	HS Transmit Data (Lane N).
TX_READY_HS_LAN*n*	Output	WORD_CLKOUT_HS	HS Transmit Ready (Lane N).

Table 71: MIPI TX D-PHY Low-Power Data Receive Mode Signals (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
TX_LPDT_ESC	Input	TX_CLK_ESC	Lane 0 Enter LPDT Mode.
TX_VALID_ESC	Input	TX_CLK_ESC	Lane 0 LPDT Data Valid .
TX_DATA_ESC [7:0]	Input	TX_CLK_ESC	Lane 0 LPDT Data Bus.
TX_READY_ESC	Output	TX_CLK_ESC	Lane 0 LPDT Data Ready.
RX_LDPT_ESC	Output	RX_CLK_ESC	Escape LP Data Receive Mode.
RX_DATA _ESC[7:0]	Output	RX_CLK_ESC	Escape Mode Receive Data.
RX_VALID_ESC	Output	RX_CLK_ESC	Escape Mode Receive Data Valid.
ERR_SYNC_ESC	Output	N/A	LPDT Data Sync Error.
RX_ULPS_ESC	Output	RX_CLK_ESC	Lane 0 entered ULPS mode.

Table 72: MIPI TX D-PHY ULP Sleep Mode (Interface to FPGA Fabric)

Signal	Direction	Clock Domain	Notes
TX_ULPS_CLK	Input	TX_CLK_ESC	CLK0 to enter Ultra-Low Power State.
TX_ULPS_EXIT	Input	TX_CLK_ESC	CLK0 to exit Ultra-Low Power State.
TX_ULPS_ACTIVE_CLK_NOT	Output	N/A	Clock Lane in ULP State - Active Low (Clk 0).
TX_ULPS_ESC_LAN*n*	Input	TX_CLK_ESC	Lane *n* to enter Ultra-Low Power State.
TX_ULPS_EXIT_LAN*n*	Input	TX_CLK_ESC	Lane *n* to exit Ultra-Low Power State.
TX_ULPS_ACTIVE_NOT_LAN*n*	Output	N/A	Data Lane in ULP State - Active Low (Lane N).

Table 73: MIPI TX D-PHY Pads

Pad	Direction	Description
MIPI*n*_TXDP[4:0]	Bidirectional	MIPI transceiver P pads.
MIPI*n*_TXDN[4:0]	Bidirectional	MIPI transceiver N pads.

Using the MIPI DPHY RX Interface

The following tables describe the settings for the Titanium MIPI DPHY RX blocks in the Interface Designer.

Table 74: Base Tab

Parameter	Choices	Notes
Instance Name	User defined	
MIPI RX Resource	None, MIPI_RX0, MIPI_RX1, MIPI_RX2, MIPI_RX3	Choose the resource.

Table 75: Control/Status Tab

Option	Choices	Notes
Calibration Clock Frequency (MHz)	80 - 120	Specify the frequency. Default: 0
<*description*> Pin Name	User defined	Control and status pin names. Efinix recommends that you use the defaults.
HS Transmit Byte/Word Clock Connection Type	gclk, rclk	Choose whether to connect to a global clock (**gclk**) or regional clock (**rclk**). Default: gclk
Invert Excape Mode Transmit Clock Pin	On or off	Turn on to invert the clock.

Table 76: Clock Lane Tab

Option	Choices	Notes
<*description*> Pin Name	User defined	Clock lane pin names. Efinix recommends that you use the defaults.

Table 77: Data Lane Tab

Option	Choices	Notes
Enable Turn-around Feature in Data Lane 0	On or off	Lane 0 can operate as a bi-directional data lane when this option is on. Default: on
Number of data lanes	1, 2, 4	Choose the number of lanes. Default: 4
Width of the data bus	8, 16	Specify the width. Default: 8
Lane *n*: Escape Mode Receive Clock Pin Name	User defined	Specify the clock.
Lane 0: Escape Mode Receive Clock Connection Type	normal, rclk	normal: Default. The clock signal is an input signal to the core. rclk: The clock signal is feeding the regional clock network.
Lane *n*: <*description*> Pin Name	User defined	Data lane pin names. Efinix recommends that you use the defaults.

Table 78: Lane Mapping Tab

Parameter	Choices	Notes
Phy Lane *n*	clk, data0, data1, data2, data3, unused	The MIPI TX block supports 4 data lanes and 1 clock lane. For each lane, specify whether to use it as clock or data. The lane mapping must be unique, which the software enforces.
Swap P&N Pin	On or off	Turn on to change which pin is P or N. This setting can be helpful when laying out your board.

Using the MIPI DPHY TX Interface

The following tables describe the settings for the Titanium MIPI DPHY TX blocks in the Interface Designer.

Table 79: Base Tab

Parameter	Choices	Notes
PHY bandwidth in Mbps	Integer up to 2500	Specify the bandwidth. Default: 2500
Instance Name	User defined	
MIPI TX Resource	None, MIPI_TX0, MIPI_TX1, MIPI_TX2, MIPI_TX3	Choose the resource.
Reference Clock Frequency	12.0, 19.2, 25.0, 26.0, 27.0, 38.4, 52.0	Choose the frequency for the reference clock.
Reference Clock Source Type	core, gpio, pll	Choose which resource generates the reference clock. For **gpio** and **pll**, the Block Editor shows you which resource to connect as the reference clock. For **core**, you specify the clock name.

Table 80: Control/Status Tab

Option	Choices	Notes
Enable Spread Spectrum Clock (SSC)	On or off	Turn on to enable SSC.
SSC Amplitude for MIPI Internal PLL (PPM)	2500 - 4999	Spread-spectrum clock amount in ppm. Default: 4999
SSC Frequency for MIPI Internal PLL (kHz)	30 - 33	Spread-spectrum clock frequency setting. Default: 30
SSC Initial Amplitude for MIPI Internal PLL (PPM)	2501 - 5000	Spread-spectrum clock initial spread down amount in ppm. The initial amplitude value must be larger than the amplitude value. Default: 5000
<*description*> Pin Name	User defined	Control and status pin names. Efinix recommends that you use the defaults.
HS Transmit Byte/Word Clock Connection Type	gclk, rclk	Choose whether to connect to a global clock (**gclk**) or regional clock (**rclk**). Default: gclk
Invert Excape Mode Transmit Clock Pin	On or off	Turn on to invert the clock.

Table 81: Clock Lane Tab

Option	Choices	Notes
Escape Mode Receive Clock Pin Name	User defined	Specify the clock name.
Escape Mode Receive Clock Connection Type	normal, rclk	normal: Default. The clock signal is an input signal to the core. rclk: The clock signal is feeding the regional clock network.
<*description*> Pin Name	User defined	Clock lane pin names. Efinix recommends that you use the defaults.

Table 82: Data Lane Tab

Option	Choices	Notes
Enable Turn-around Feature in Data Lane 0	On or off	Lane 0 can operate as a bi-directional data lane when this option is on. Default: on
Number of data lanes	1, 2, 4	Choose the number of lanes. Default: 4
Width of the data bus	8, 16	Specify the width. Default: 8
<*description*> Pin Name	User defined	Data lane pin names. Efinix recommends that you use the defaults.

Table 83: Lane Mapping Tab

Parameter	Choices	Notes
Phy Lane *n*	clk, data0, data1, data2, data3, unused	The MIPI TX block supports 4 data lanes and 1 clock lane. For each lane, specify whether to use it as clock or data. The lane mapping must be unique, which the software enforces.
Swap P&N Pin	On or off	Turn on to change which pin is P or N. This setting can be helpful when laying out your board.

Table 84: Timing Tab

Efinix encourages you to maintain the current settings of the timing parameter.

Parameter	Choices	Notes
T_{CYCLE_SEL}	0 - 7	Internal register value.
$T_{PLL\_FBK\_FRA}$	0 - 16777215	Internal register value.
$T_{PLL\_FBK\_INT}$	0 - 511	Internal register value.
$T_{PLL\_PRE\_DIV}$	0 - 3	Internal register value.
$T_{CLANE\_HS\_CLK\_POST\_TIME}$	0 - 255	Internal register value.
$T_{CLANE\_HS\_CLK\_PRE\_TIME}$	0 - 255	Internal register value.
$T_{CLANE\_HS\_PRE\_TIME}$	0 - 255	Internal register value.
$T_{CLANE\_HS\_TRAIL\_TIME}$	0 - 255	Internal register value.
$T_{CLANE\_HS\_ZERO\_TIME}$	0 - 255	Internal register value.
$T_{DLANE\_HS\_PRE\_TIME}$	0 - 255	Internal register value.
$T_{DLANE\_HS\_TRAIL\_TIME}$	0 - 255	Internal register value.
$T_{DLANE\_HS\_ZERO\_TIME}$	0 - 255	Internal register value.

Design Check: MIPI DPHY Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

mipi_dphy_rule_word_clk (error)

Message	Empty byte/word clock pin name
To fix	You need to specify the **HS Transmit Byte/Word Clock Pin Name** in the **Control/Status** tab.

mipi_dphy_rule_clock_lane (error)

Message	No clock lane was assigned
To fix	You need to choose a clock lane.

mipi_dphy_rule_inst_name (error)

Message	Instance name is empty
To fix	Enter the instance name.

mipi_dphy_rule_lane_num (error)

Message	Number of data lanes <#> doesn't match with Lane assignment count
To fix	The number of lanes used must match the number of lanes you specified.

mipi_dphy_rule_resource (error)

Message	Resource name is empty
To fix	Specify a MIPI D-PHY TX or RX resource.

Message	Resource <name> is not a valid MIPI <RX/TX> device instance
To fix	The resource you specified does not exist. Check whether you have a typo in the resource name.

mipi_dphy_rule_tx_ref_clk_resource (error)

Message	Reference clock resource <res_name> is not configured as mipi_clkin connection
To fix	You get this error when the reference clock resource (GPIO block) is set to **gpio** instead of **mipi_clkin**. Change the setting for the GPIO block.

Message	Reference clock resource <res_name> input name is empty
To fix	Specify the name.

Message	Reference clock resource <res_name> is not configured as input
To fix	The GPIO block you are using as the reference clock must be configured as an input. Change the **Mode** setting for the GPIO block.

Message	Reference clock GPIO resource <res_name> has not been configured
To fix	You added the GPIO block but did not configure it correctly. It should be an input and should be configured as **Connection Type** > **mipi_clkin**.

Message	Reference clock PLL resource <res_name> Output Clock <#> has not been configured
To fix	If you choose **Reference Clock Source Type** > **pll**, you also need to configure the output clock for the PLL resource shown in the **Base** tab.

Message	Reference clock PLL resource <res_name> has not been configured
To fix	If you choose **Reference Clock Source Type** > **pll**, you also need to configure the PLL resource shown in the **Base** tab.

Message	Core reference clock pin name is empty
To fix	If you choose **Reference Clock Source Type** > **core**, then you need to enter the name for the pin.

mipi_dphy_rule_rx_cfg_clk (error)

Message	Calibration Clock Frequency range: 80-120 MHz
To fix	The setting for **Control/Status** > **Configuration Clock Frequency (MHz)** is not in the correct range. Pick a number between 80 and 120.

mipi_dphy_rule_tx_phy_clk (warning)

Message	PHY Bandwidth has to be in steps of 10 Mbps
To fix	You need to use bandwidth values in 10s for the PHY bandwidth. For example, use 2500 and not 2495.

mipi_dphy_rule_tx_pll_ssc_init_amp (error)

Message	SSC Amplitude must be less than SSC Init Amplitude
To fix	Change the values so that the SSC Amplitude is less than SSC Init Amplitude.

Chapter 11

PLL Interface

Contents:

Efinix FPGAs have one or more PLLs to generate clock frequencies. The number of PLLs and the PLL features vary depending on the family and FPGA.

Table 85: PLL Types

PLL Version	Description	Available In
PLL V3	Full-featured PLL	Titanium Ti35, Ti60, Ti90, Ti120, and Ti180 FPGAs.
FPLL V1	Fractional PLL	Titanium Ti85, Ti135, Ti165, Ti240, and Ti375 FPGAs.
PLL SSC	Spread-spectrum clocking PLL	Titanium FPGAs that have a hardened MIPI D-PHY interface.

Note: V*x* refers to the version number. This number is used to reference the PLL in the Efinity Python API and interface primitives.

About the PLL Interface

Titanium FPGAs have PLLs to synthesize clock frequencies. The PLLs are located in the corners of the FPGA. You can use the PLL to compensate for clock skew/delay via external or internal feedback to meet timing requirements in advanced applications. The PLL reference clock has up to four sources. You can dynamically select the PLL reference clock with the `CLKSEL` port. (Hold the PLL in reset when dynamically selecting the reference clock source.)

The PLL consists of a pre-divider counter (N counter), a feedback multiplier counter (M counter), a post-divider counter (O counter), and output dividers (C).

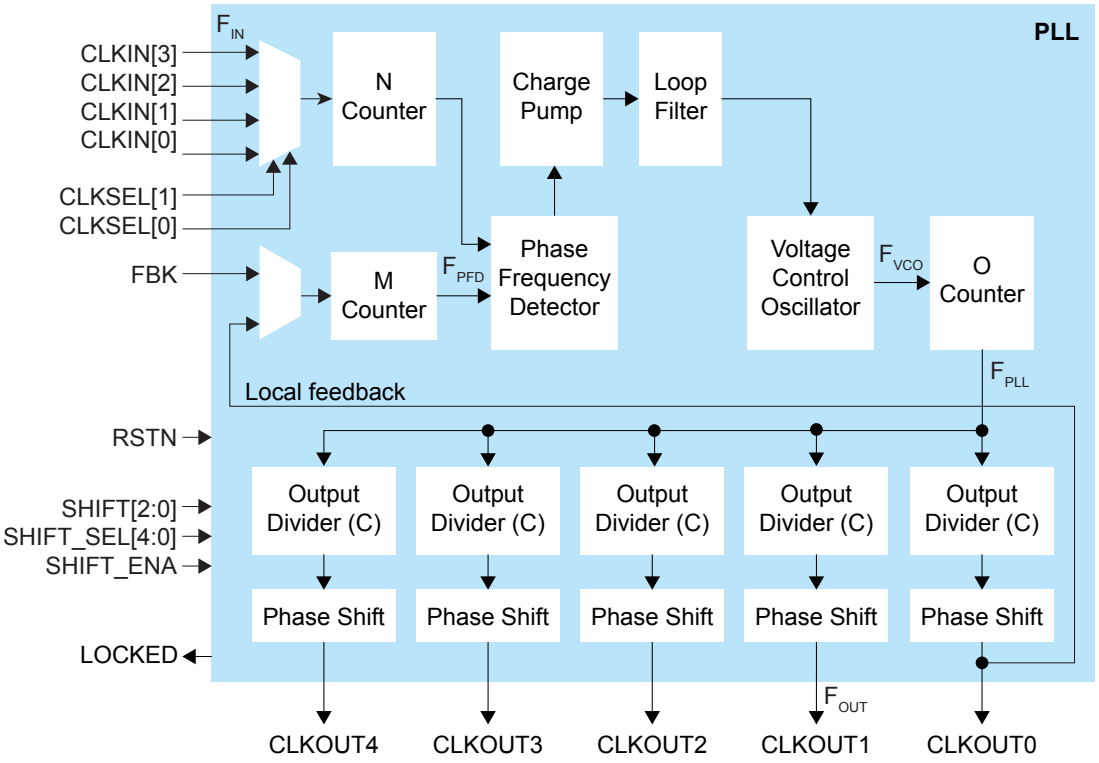
At startup, Efinix recommends that you hold the PLL in reset until the PLL's reference clock source is stable.

Note: You can cascade PLLs. To avoid the PLL losing lock, Efinix recommends that you do not cascade more than two PLLs.

At startup, Efinix recommends resetting all cascaded PLLs. Hold the first PLL in reset until the PLL's reference clock source is stable. Hold the cascaded PLLs in reset until the previous PLL is locked.

Cascaded PLLs do not need a 50% duty cycle on the reference clock. However, the clock needs to meet the PLL minimum pulse width as specified in the data sheet.

Figure 47: PLL Block Diagram



The counter settings define the PLL output frequency:

Local and Core Feedback Mode	Where:
$F_{PFD} = F_{IN} / N$ $F_{VCO} = (F_{PFD} \times M \times O \times C_{FBK})$ [14] $F_{PLL} = F_{VCO} / O$ $F_{OUT} = (F_{IN} \times M \times C_{FBK}) / (N \times C)$	F_{VCO} is the voltage control oscillator frequency F_{PLL} is the post-divider PLL VCO frequency F_{OUT} is the output clock frequency F_{IN} is the reference clock frequency F_{PFD} is the phase frequency detector input frequency O is the post-divider counter C is the output divider

Note: Refer to the PLL Timing and AC Characteristics for F_{VCO}, F_{OUT}. F_{IN}, F_{PLL}, and F_{PFD} values.

[14] $(M \times O \times C_{FBK})$ must be ≤ 255.

Figure 48: PLL Interface Block Diagram

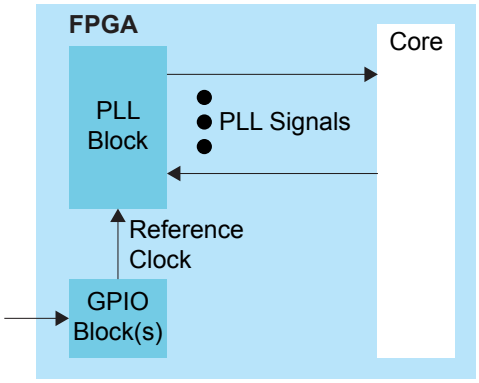


Table 86: PLL Signals (Interface to FPGA Fabric)

Signal	Direction	Description
CLKIN[3:0]	Input	Reference clocks driven by I/O pads or core clock tree.
CLKSEL[1:0]	Input	You can dynamically select the reference clock from one of the clock in pins.
RSTN	Input	Active-low PLL reset signal. When asserted, this signal resets the PLL; when de-asserted, it enables the PLL. De-assert only when the CLKIN signal is stable. Connect this signal in your design to power-up or reset the PLL. Assert the RSTN pin for a minimum pulse of 10 ns to reset the PLL. Assert RSTN when dynamically changing the selected PLL reference clock.
FBK	Input	Connect to a clock out interface pin when the PLL is in when the PLL is not in internal feedback mode.
CLKOUT0 CLKOUT1 CLKOUT2 CLKOUT3 CLKOUT4	Output	PLL output. You can route these signals as input clocks to the core's GCLK network. **Ti35, Ti60:** CLKOUT4 can only feed the top or bottom regional clocks. **Ti35, Ti60:** All PLL outputs lock on the negative clock edge. The Interface Designer inverts the clock polarity on the leaf cells by default (Invert Output Clock option unchecked). Check the option if you are using the clock to drive core logic. You can use CLKOUT0 only for clocks with a maximum frequency of 4x (integer) of the reference clock. If all your system clocks do not fall within this range, you should dedicate one unused clock for CLKOUT0.
LOCKED	Output	Goes high when PLL achieves lock; goes low when a loss of lock is detected. Connect this signal in your design to monitor the lock status. This signal is not synchronized to any clock and the minimum high or low pulse width of the lock signal may be smaller than the CLKOUT's period.
SHIFT[2:0]	Input	(Optional) Dynamically change the phase shift of the output selected to the value set with this signal. Possible values from 000 (no phase shift) to 111 (3.5 F_{PLL} cycle delay). Each increment adds 0.5 cycle delay.
SHIFT_SEL[4:0]	Input	(Optional) Choose the output(s) affected by the dynamic phase shift.
SHIFT_ENA	Input	(Optional) When high, changes the phase shift of the selected PLL(s) to the new value.

Learn more: Refer to the device data sheet for the list of PLL reference clock assignments.

Table 87: PLL Interface Designer Settings

Parameter	Choices	Notes
Instance Name	User defined	
PLL Resource		The resource listing depends on the FPGA you choose.
Output Clock Inversion	On, off	Turn on to invert each output clock individually.
Connection Type	gclk, rclk	Output clocks 3 and 4 can feed the global clock network (default). Some PLLs can also drive the regional clock network; see "Driving the Regional Network" in the data sheet for details.
Clock Source	External	PLL reference clock comes from an external pin. Select the available external clock.
	Dynamic	PLL reference clock comes from four possible sources (external and core), and are controlled by the clock select bus. Specify the clock selector pin name and core clock pin name
	Core	PLL reference clock comes from the core. Specify the core clock pin name.
Automated Clock Calculation		Pressing this button launches the PLL Clock Calculation window. The calculator helps you define PLL settings in an easy-to-use graphical interface. Refer to **Using the PLL Clock Calculator** on page 150 for more details.

Using the PLL Block

The Titanium FPGA's PLL block lets you configure the reference clock, feedback options, frequency, and output clocks for the PLL. This PLL is referenced as PLL_V3 in the Python API. You set up the PLL using the PLL Clock Calculator, which provides an easy-to-use graphical way to specify the frequencies and other settings.

- In the PLL's **Properties** tab, you specify general settings such as the instance name, PLL resource, clock source, and external clock.
- You can invert any of the clock outputs by clicking **Inverted** for the clock output in the **Output Clock Inversion** box.
- Click the **Automated Clock Calculation** button to open the PLL Clock Calculator.

Reference Clock Settings

The PLL has four possible reference clocks. Depending on the PLL, one or two of the clocks can come from the FPGA core, and two or three can come from off chip. You select the clocks using the **Clock Source** drop-down box:

- **core**—The PLL reference clock comes from the FPGA core.
- **external**—Enables clock 0, 1, or 2. The PLL reference clock comes from an external pin. The GUI displays the resource(s) that can be the reference clock.

  Note: In this mode, a GPIO block with a **pll_clkin** connection type must generate the reference clock(s). The software displays which resource(s) you can use (and the instance name if you have created it).

  1. Add a GPIO block.
  2. Enter the instance name.
  3. Choose **input** as the mode.
  4. Choose **pll_clkin** as the connection type.
  5. In the Resource Assigner, assign it to the resource shown in the PLL's Properties tab.
- **dynamic**—Enables all four clocks; requires a clock selector bus to choose the clock dynamically. The GUI displays the resource(s) that can be the reference clock.

Using the PLL Clock Calculator

The PLL Clock Calculator provides a graphical way for you to set up the advanced PLL block. When you open the calculator, the GUI appears in automatic mode, which provides basic settings. You can:

- Choose the feedback mode (**Local**, **Core**, or **External**).
- Turn signals on (gray x) or off (green arrow) by clicking the icons next to the signal.
- Specify the signal names.
- Specify the phase shift.
- Choose which clock has feedback (for core feedback mode).

The Titanium PLL supports dynamic phase shifting. To enable it, click the **Dynamic** button for the clock output. The calculator adds three additional pins that you use to control the dynamic shifting. You need to specify the pin names.

As you make selections, the calculator determines the values for the pre-divider, multiplier, post divider, and clock dividers that meet your settings. The GUI prompts you if you make selections that are impossible to solve.

In manual mode, the interface displays the PLL's internal block diagram, and provides boxes for you to set the values for the pre-divider, multiplier, post divider, and clock dividers. As you adjust the values, the calculator prompts you if you make settings that result in F_{VCO} values that are out of range or are impossible to solve. When you turn manual mode off,

the calculator adjusts the output clock frequencies to match the manual settings. If you have incorrect settings for the pre-divider, multiplier, post divider, and clock dividers, when you turn manual mode off, the calculator adjusts the values to ones that allow a valid solution.

When you are finished using the calculator, click **Finish** to save your settings and close the GUI.

Manually Configuring the PLL

If you want more control over the PLL, you can manually configure it in the PLL Clock Calculator using manual mode. To enable this mode, click the **Manual Mode** slider to turn it on.

Table 88: PLL Manual Mode Options

Option	Choices	Description
Pre-Divider (N)	1, 2, 4	The pre-divider value.
Multiplier (M)	1, 2, 4	The multiplier value.
Post-Divider (O)	1, 2, 4, 8, 16, 32, 64, 128	Post-divider value.
CLK Divider *n*	2 - 128	Clock divider for each output.
Phase Shift Value *n*	0 - 7	Post-divider VCO cycle delay to phase shift, in degrees: 0: No phase shift 1: 0.5 2: 1 3: 1.5 4: 2 5: 2.5 6: 3 7: 3.5

The post-divider VCO cycle delay relates to the phase shift as shown in the following equation:

Phase shift (in degrees) = ((Post-divider VCO cycle delay x O x F_{CLKOUT}) / F_{VCO}) x 360

where:

- O is the post-VCO division ration
- F_{CLKOUT} is the output frequency
- F_{VCO} is the VCO frequency

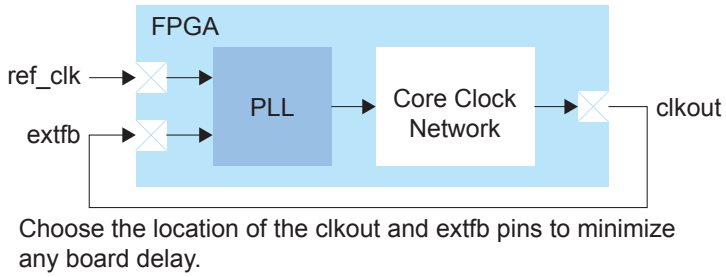
Implementing a Zero-Delay Buffer

Titanium PLLs can implement a zero-delay buffer circuit. In this mode, the PLL removes all of the clock-insertion delay from the input I/O buffer and core clock tree. You may want to use this buffer when you have a single clock signal that fans out to more than one destination with low skew.

To implement a zero-delay buffer:

- In the PLL Clock Calculator, use **Feedback Mode > External**.
- Add a GPIO for the clock output.
- Add a second GPIO for the external feedback.
- Add a third GPIO for the PLL reference clock.
- On your board, connect the clock output pin to the external feedback pin.

Figure 49: Zero-Delay Buffer Block Diagram



The following code example shows an .isf that implements a zero-delay buffer for the Ti60 F225.

```
# Efinity Interface Configuration
# Version: 2021.1.165.2.19
# Date: 2021-09-23 15:23
#
# Copyright (C) 2017 - 2021 Efinix Inc. All rights reserved.
#
# Device: Ti60F225
# Package: 225-ball FBGA (preliminary)
# Project: r4000
# Configuration mode: active (x1)
# Timing Model: C4 (preliminary)

# Create instance
design.create_clockout_gpio("clkout")
design.create_pll_ext_fb_gpio("fbk_clk")
design.create_pll_input_clock_gpio("ref_clk")
design.create_block("pll_inst1","PLL")

# Set property, non-defaults
design.set_property("clkout","OUT_CLK_PIN","clk")
design.set_property("pll_inst1","CLKOUT0_EN","1","PLL")
design.set_property("pll_inst1","CLKOUT1_EN","0","PLL")
design.set_property("pll_inst1","CLKOUT2_EN","0","PLL")
design.set_property("pll_inst1","CLKOUT3_EN","0","PLL")
design.set_property("pll_inst1","CLKOUT4_EN","0","PLL")
design.set_property("pll_inst1","REFCLK_SOURCE","EXTERNAL","PLL")
design.set_property("pll_inst1","CLKOUT0_DIV","82","PLL")
design.set_property("pll_inst1","CLKOUT0_DYNPHASE_EN","0","PLL")
design.set_property("pll_inst1","CLKOUT0_PHASE_SETTING","0","PLL")
design.set_property("pll_inst1","CLKOUT0_PIN","clk","PLL")
design.set_property("pll_inst1","EXT_CLK","EXT_CLK0","PLL")
design.set_property("pll_inst1","LOCKED_PIN","","PLL")
design.set_property("pll_inst1","M","1","PLL")
3design.set_property("pll_inst1","N","1","PLL")
4design.set_property("pll_inst1","O","2","PLL")
design.set_property("pll_inst1","OUTPUT_CLOCKS_INVERTED","0","PLL")
design.set_property("pll_inst1","PHASE_SHIFT_ENA_PIN","","PLL")
design.set_property("pll_inst1","PHASE_SHIFT_PIN","","PLL")
design.set_property("pll_inst1","PHASE_SHIFT_SEL_PIN","","PLL")
design.set_property("pll_inst1","REFCLK_FREQ","33.33","PLL")
design.set_property("pll_inst1","RSTN_PIN","","PLL")
design.set_property("pll_inst1","FEEDBACK_MODE","EXTERNAL","PLL")
design.set_property("pll_inst1","FEEDBACK_CLK","CLK0","PLL")

# Set resource assignment
design.assign_pkg_pin("clkout","M7")
8design.assign_pkg_pin("ref_clk","P2")
9design.assign_pkg_pin("fbk_clk","R6")
10design.assign_resource("pll_inst1","PLL_BL0","PLL")
```

Design Check: PLL Messages

Note: These design rules are also applicable to fractional PLL blocks.

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

pll_rule_clksel_pin (error)

Message	Valid characters are alphanumeric characters with dash and underscore only.
To fix	Update the pin name for the clock selector pin when reference clock mode is dynamic

pll_rule_dynamic_shift_invalid_pin (error)

Message	Invalid pin names found: <pin names>
To fix	Specify a valid pin name.

pll_rule_dynamic_shift_feedback (error)

Message	Output clock <name> used as feedback cannot be set with dynamic phase shift
To fix	If you are using a PLL output clock for feedback, you cannot use dynamic phase shifting. Instead, specify a phase.

pll_rule_dynamic_shift_empty_pin (error)

Message	Dynamic phase shift is enabled but missing pin names: <list>
To fix	For a PLL output, if you are turn on **Dynamic** for the phase shift, you also need to specify names for the SHIFT, SHIFT_SELECT, and SHIFT_ENA pins.

pll_rule_extfb_io (error)

Message	External feedback and reference clock have to be of the same instance type and IO standard
To fix	Use the same I/O standard for the external feedback clock and reference clock GPIO blocks.

pll_rule_extfb_resource (error)

Message	There can only be one configured resource for external IO feedback
To fix	In external feedback mode, you can only specify one clock out pin for feedback.
Message	External IO feedback resource <name> is not configured as pll_extfb connection
To fix	In the GPIO block that is your feedback resource, set the **Connection Type** to **pll_extfb**.
Message	The resource for external IO feedback is not configured
To fix	Add a GPIO block in **input** mode, set the **Connection Type** to **pll_extfb**, and assign it to a resource that supports the pll_extb connection type.
Message	External IO feedback resource is unbonded
To fix	Use a resource that is available in the device for external I/O feedback.

pll_rule_fb_freq (error)

Message	Feedback frequency <#>MHz is out of range. Min=<>MHz Max=<>MHz
To fix	The feedback frequency needs to be within the range specified. Adjust the parameters to meet that requirement.

pll_rule_feedback_clock (error)

Message	Feedback clock name is required with non-internal feedback
To fix	You need to specify a clock pin name when you are not using internal feedback mode.

Message	Feedback clock name <string> is not from the same PLL
To fix	You need to use one of the output clocks from the PLL you are configuring as the feedback clock. You cannot use an output clock from a different PLL.

Message	Feedback clock in local mode has to connect to output clock 0
To fix	When **Feedback Mode** is **Local**, you can only use output clock 0 for feedback.

Message	Feedback clock <string> is not connected to pll clkout
To fix	The feedback clock you are using needs to be one of the output clocks from the PLL.

pll_rule_feedback_clock (info)

Message	Feedback clock phase shift is not 0-degree, check that the feedback clock is in-phase with the reference clock.
To fix	Set the feedback clock phase to 0 degrees in the PLL Clock Calculator. Efinix recommends a 0 degree phase for feedback clocks.

pll_rule_feedback_clk_regional (error)

Message	Unroutable regional clock output 4 to the core feedback interface with external reference clock resource set to <resource name>. Select a different reference clock resource or assign a different output clock as feedback clock.
To fix	This error only applies to Ti35, Ti60 FPGAs. Use a different I/O resource as the external reference clock or use a different output clock as the feedback clock.

pll_rule_feedback_mode (error)

Message	Internal feedback mode is not supported
To fix	You may receive this error when configuring an interface with the API. Do not use internal feedback mode as it's not supported.

pll_rule_input_freq (error)

Message	Input Frequency <float> MHz (after pre-divider) is out of range. Min=<float>MHz Max=<float>MHz
To fix	Assign the reference clock frequency to a value within the specified range.

pll_rule_input_freq_limit (error)

Message	Input Frequency <float> MHz is out of range. Min=<float>MHz Max=<float>MHz
To fix	Assign the right reference clock frequency.

pll_rule_inst_name (error)

Message	Instance name is empty Valid characters are alphanumeric characters with dash and underscore only
To fix	Specify a valid instance name.

pll_rule_mipi_tx_clock (error)

Message	PLL output clock <name> is not allowed to connect to MIPI TX Lane Serial and Parallel clocks at the same time
To fix	You cannot use the slow clock for one MIPI TX lane as the fast clock for a different MIPI TX lane.

pll_rule_multiplier (error)

Message	Multiplier is invalid. Valid values are <*values*>.
To fix	The multiplier frequency needs to be within the range specified. Adjust the parameters to meet that requirement.

pll_rule_oc_cascade (warning)

Message	(O * C == 1), PLL cascading is not supported with possible PLL clkout: <clock names>
To fix	The listed PLL output clocks cannot be cascaded. Change your design so they are not cascaded.

pll_rule_out_clk_conn_type (error)

Message	<PLL resource> does not support regional clock connection on output clock #
To fix	Some FPGAs allow you to choose whether to connect output clocks 3 and 4 to the global or regional clock network. You get this error when the PLL you selected does not support the regional clock connection. Change the setting to use the global clock or use another PLL.

pll_rule_outclk_div_pshift (warning)

Message	Enable phase shift with odd output clock divider will result in duty cycle distortion on the output clock. It is not advisable to use the clock for double data operation: <output clock names>
To fix	You can ignore this warning message if you are not using the clock output for double data rates. If you are using double data rate, instead choose an even clock divider.

pll_rule_output_clock (error)

Message	At least one output clock must be configured
To fix	Configure at least one PLL output clock and specify the output clock pin name.

Message	Output clock count is out of range. Min=<int>Max=<int>
To fix	You have specified the wrong number of output clocks (too many or none).

pll_rule_output_divider (error)

Message	Output divider for <clock name> is invalid. Valid values are between 1-128
To fix	Choose a value between 1 and 128.

pll_rule_output_freq (error)

Message	Output frequency <float>MHz is out of range. Min=<float>MHz Max=<float>MHz
To fix	The output frequency needs to be within the range specified. Adjust the parameters to meet that requirement.

pll_rule_output_name (error)

Message	PLL output clock names have to be unique. Duplicates found: <list of string>
To fix	You get this error when you use duplicate clock names. Rename them.

pll_rule_output_number (error)

Message	Output number for <clk name> is invalid. It must be between 0 to <int>
To fix	The output clocks are numbered (e.g., CLKOUT3). Make sure that the number is within specified range.

pll_rule_param (error)

Message	Invalid parameters configuration: <feature>
To fix	Performs a general check for invalid parameters. Review the other error messages.

pll_rule_pll_freq (error)

Message	PLL Frequency is out of range, Freq = <value> Min = <min> MHz Max = <max> MHz
To fix	The maximum post-divided VCO clock fmax is 4,000 Mhz. Change the PLL clock calculator settings so that it is in range.

pll_rule_post_divider (error)

Message	Post-divider is invalid. Valid values are <list of int>
To fix	Choose a post divider value from the list shown.

pll_rule_pre_divider (error)

Message	Pre-divider is out of range. Min=<int> Max=<int>
To fix	The pre-divider frequency needs to be within the range specified. Adjust the parameters to meet that requirement.

pll_rule_refclk (error)

Message	Bonded external reference clock pin has to be specified in dynamic mode
To fix	When using **dynamic** as the **Clock Source**, the PLL expects to find the resource for the external clock(s). Add a GPIO block in **input** mode, set the **Connection Type** to **pll_clkin**, and assign it to the resource shown in the PLL **Properties** tab under **Dynamic Clock**.

Message	Both core refclk pins have to be specified in dynamic mode
To fix	When using **dynamic** as the **Clock Source**, you need to specify the names for the core clocks 0 and 1. (Some PLLs use 2 core clocks in dynamic mode.)

Message	Core refclk pin has to be specified in core mode
To fix	When using **core** as the **Clock Source**, you need to specify the pin name.

Message	Core refclk pin has to be specified in dynamic mode
To fix	When using **dynamic** as the **Clock Source**, you need to specify the names for core clock 0. (Some PLLs use only 1 core clock in dynamic mode.)

Message	External refclk pin has to be set in external mode
To fix	When using **external** as the **Clock Source**, the PLL expects to find the resource for the external clock. Add a GPIO block in **input** mode, set the **Connection Type** to **pll_clkin**, and assign it to the resource shown in the PLL **Properties** tab under **External Clock**.

Message	External reference clock resource {} is not configured as pll_clkin connection
To fix	You use a GPIO block configured in alternate connection mode to be the reference clock for the PLL. Change the GPIO **Connection Type** to **pll_clkin**.

Message	Reference clock at <resource> connected to external clock pin {0|1|2} has not been defined
To fix	The PLL expects to find the resource for the external clock. Add a GPIO block in **input** mode, set the **Connection Type** to **pll_clkin**, and assign it to the resource shown in the PLL **Properties** tab under **External Clock**.

Message	Invalid external clock {0|1|2} resource selected: Resource Unbonded
To fix	In the FPGA/package combination you are using, you cannot use the external clock resoure specified because it is not available in the package.

Message	The resource for CLKIN[<index>] is not configured
To fix	The PLL expects to find the resource for the PLL clockin. Add a GPIO block in **input** mode, set the **Connection Type** to **pll_clkin**, and assign it to the correct resource.

Message	There can only be one configured resource for CLKIN[<index>]
To fix	Each clock input can have only one resource. The software issues an internal error. Contact your local FAE for help.

pll_rule_resource (error)

Message	Resource name is empty Resource is not a valid PLL device instance
To fix	Choose a valid PLL resource.

pll_rule_vco_freq (error)

Message	VCO frequency is out of range. Freq=<float> Min=<float>MHz Max=<float>MHz
To fix	The VCO frequency needs to be within the range specified. Adjust the parameters to meet that requirement.

About the Fractional PLL Interface

Titanium FPGAs have PLLs to synthesize clock frequencies. The PLLs are located in the corners of the FPGA. You can use the PLL to compensate for clock skew/delay via external or internal feedback to meet timing requirements in advanced applications. The PLL reference clock has up to four sources. You can dynamically select the PLL reference clock with the `CLKSEL` port. (Hold the PLL in reset when dynamically selecting the reference clock source.)

Titanium FPGAs also support dynamic reconfiguration, programmable duty cycle, a fractional output divider, and spread-spectrum clocking. These features are described in later sections. The PLL consists of a pre-divider counter (N counter), a feedback multiplier counter (M counter), a post-divider counter (O counter), and output dividers (C). A delta sigma modulator supports the fractional output divider features.

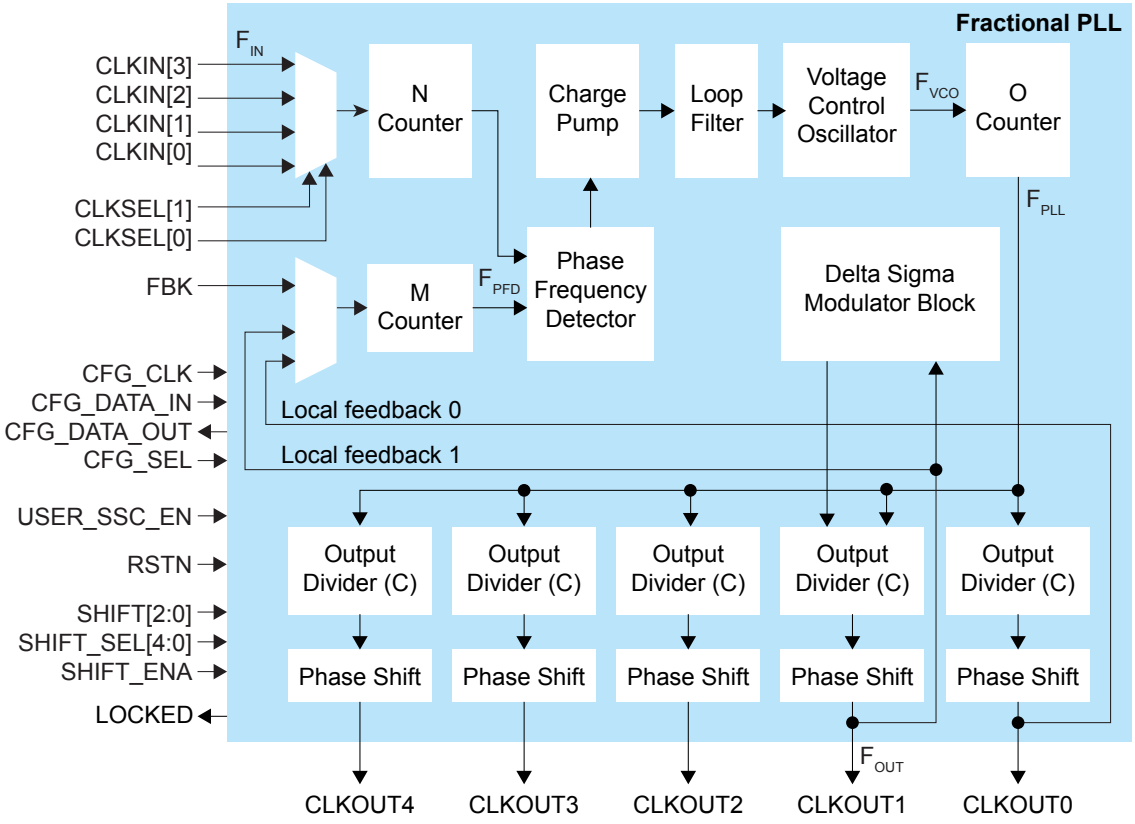
At startup, Efinix recommends that you hold the PLL in reset until the PLL's reference clock source is stable.

Note: You can cascade the PLLs in Titanium FPGAs. To avoid the PLL losing lock, Efinix recommends that you do not cascade more than two PLLs.

At startup, Efinix recommends resetting all cascaded PLLs. Hold the first PLL in reset until the PLL's reference clock source is stable. Hold the cascaded PLLs in reset until the previous PLL is locked.

Cascaded PLLs do not need a 50% duty cycle on the reference clock. However, the clock needs to meet the PLL minimum pulse width as specified in the data sheet.

Figure 50: PLL Block Diagram



The counter settings define the PLL output frequency:

Feedback Mode	Where:
$F_{PFD} = F_{IN} / N$ $F_{VCO} = (F_{PFD} \times M \times O \times C_{FBK})$ $F_{PLL} = F_{VCO} / O$ $F_{OUT} = (F_{IN} \times M \times C_{FBK}) / (N \times C)$	F_{VCO} is the voltage control oscillator frequency F_{PLL} is the post-divider PLL VCO frequency F_{OUT} is the output clock frequency F_{IN} is the reference clock frequency F_{PFD} is the phase frequency detector input frequency O is the post-divider counter C is the output divider

Note: Refer to the PLL Timing and AC Characteristics for F_{VCO}, F_{OUT}. F_{IN}, F_{PLL}, and F_{PFD} values.

Figure 51: PLL Interface Block Diagram

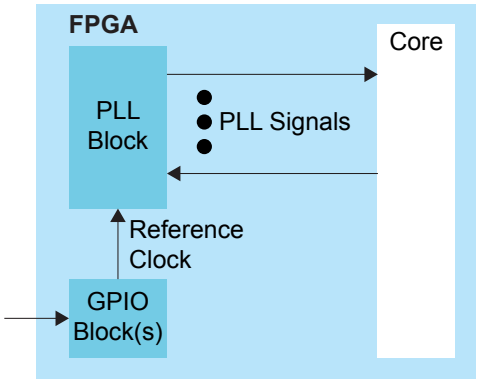


Table 89: PLL Signals (Interface to FPGA Fabric)

Signal	Direction	Description
CLKIN[3:0]	Input	Reference clocks driven by I/O pads or core clock tree. In dynamic mode, the CLKSEL pin chooses which of these inputs to use.
CLKSEL[1:0]	Input	You can dynamically select the reference clock from one of the clock in pins.
RSTN	Input	(Optional) Active-low PLL reset signal. When asserted, this signal resets the PLL; when de-asserted, it enables the PLL. De-assert only when the CLKIN signal is stable. Connect this signal in your design to power-up or reset the PLL. Assert the RSTN pin for a minimum pulse of 10 ns to reset the PLL. Assert RSTN when dynamically changing the selected PLL reference clock.
FBK	Input	Connect to a clock out interface pin when the PLL is not in internal feedback mode. Required when any output is using dynamic phase shift.
CLKOUT0 CLKOUT1 CLKOUT2 CLKOUT3 CLKOUT4	Output	PLL output. You can route these signals as input clocks to the core's GCLK network. The PLL output clock used as the feedback clock can have a maximum frequency of 4x (integer) of the reference clock. If all your system clocks do not fall within this range, you should dedicate one unused PLL output clock for feedback.
LOCKED	Output	(Optional) Goes high when PLL achieves lock; goes low when a loss of lock is detected. Connect this signal in your design to monitor the lock status. This signal is not synchronized to any clock and the minimum high or low pulse width of the lock signal may be smaller than the CLKOUT's period.
SHIFT[2:0]	Input	(Optional) Dynamically change the phase shift of the output selected to the value set with this signal. Possible values from 000 (no phase shift) to 111 (3.5 F_{PLL} cycle delay). Each increment adds 0.5 cycle delay. Required when any output is using dynamic phase shift.
SHIFT_SEL[4:0]	Input	(Optional) Choose the output(s) affected by the dynamic phase shift. Required when any output is using dynamic phase shift.
SHIFT_ENA	Input	(Optional) When high, changes the phase shift of the selected PLL(s) to the new value. Required when any output is using dynamic phase shift.
CFG_CLK	Input	Configuration clock pin name; used with dynamic configuration.

Signal	Direction	Description
CFG_DATA_IN	Input	Configuration data input pin name; used with dynamic configuration.
CFG_DATA_OUT	Output	Configuration data output pin name; used with dynamic configuration.
CFG_SEL	Input	Configuration select pin name; used with dynamic configuration.
USER_SSC_EN	Input	User spread-spectrum clocking enable pin name.

Learn more: Refer to the device data sheet for the list of PLL reference clock assignments.

Table 90: PLL Interface Designer Settings

Parameter	Choices	Notes
Instance Name	User defined	
PLL Resource		The resource listing depends on the FPGA you choose.
Output Clock Inversion	On, off	Turn on to invert each output clock individually.
Connection Type	gclk, rclk	Output clocks 3 and 4 can feed the global clock network (default). Some PLLs can also drive the regional clock network; see "Driving the Regional Network" in the data sheet for details.
Clock Source	External	PLL reference clock comes from an external pin. Select the available external clock.
	Dynamic	PLL reference clock comes from four possible sources (external and core), and are controlled by the clock select bus. Specify the clock selector pin name and core clock pin name
	Core	PLL reference clock comes from the core. Specify the core clock pin name.
Automated Clock Calculation		Pressing this button launches the PLL Clock Calculation window. The calculator helps you define PLL settings in an easy-to-use graphical interface. Refer to Using the Fractional PLL Clock Calculator on page 164 for details.

Using the Fractional PLL Block

Titanium FPGA's PLL block lets you configure the reference clock, feedback options, frequency, and output clocks for the PLL. You set up the PLL using the PLL Clock Calculator, which provides an easy-to-use graphical way to specify the frequencies and other settings.

- In the PLL's **Properties** tab, you specify general settings such as the instance name, PLL resource, clock source, and external clock.
- You can invert any of the clock outputs by clicking **Inverted** for the clock output in the **Output Clock Inversion** box.
- Clock outputs 3 and 4 can feed the global or regional clock network. Choose **gclk** (default)) or **rclk** under **Connection Type**.
- Click the **Automated Clock Calculation** button to open the PLL Clock Calculator.

Reference Clock Settings

The PLL has four possible reference clocks. Depending on the PLL, one or two of the clocks can come from the FPGA core, and two or three can come from off chip. You select the clocks using the **Clock Source** drop-down box:

- **core**—The PLL reference clock comes from the FPGA core.

- **external**—Enables clock 0 or 1. The PLL reference clock comes from an external pin. The GUI displays the resource(s) that can be the reference clock.

  Note: In this mode, a GPIO block with a **pll_clkin** connection type must generate the reference clock(s). The software displays which resource(s) you can use (and the instance name if you have created it).

  1. Add a GPIO block.
  2. Enter the instance name.
  3. Choose **input** as the mode.
  4. Choose **pll_clkin** as the connection type.
  5. In the Resource Assigner, assign it to the resource shown in the PLL's Properties tab.
- **dynamic**—Enables all four clocks; requires a clock selector bus to choose the clock dynamically. The GUI displays the resource(s) that can be the reference clock.

Advance Setting Tab: SSC

The fractional PLL supports spread-spectrum clocking. The available modes are:

- **Disable**—Turn off spread-spectrum clocking.
- **Static**—Always use spread-spectrum clocking.
- **Dynamic**—Only use spread-spectrum clocking when activated by an enable signal.

Table 91: Spread Spectrum Clocking Options

Parameter	Choices	Description
SSC Modulation Frequency (KHz)	User defined	Enter the frequency.
SSC Modulation Amplitude (%)	User defined	Enter the amplitude.
SSC Spread Direction	Up, Down (default), Center	Indicate how the spectrum is applied to the waveform.
User SSC Enable Pin Name	User defined	Indicate the enable pin name. Dynamic mode only.

Note: To use SSC, you must also enable fractional feedback mode (see Programmable Duty Cycle and Fractional Feedback on page 164).

Advance Setting Tab: Dynamic Reconfiguration

The fractional PLL supports a dynamic reconfiguration mode, which lets you change the PLL setting on the fly after initial boot-up without having to re-configure the FPGA. Turn on **Enable** to use this feature.

You need to enable the reset and lock signals !!!

Efinix provides the PLL Dynamic Reconfiguration core and the PLL Dynamic Reconfiguration Wizard to help you integrate this feature in your design.

Important: Refer to the PLL Dynamic Reconfiguration Core User Guide for details on how to use this feature and how to integrate the PLL Dynamic Reconfiguration Core and Interface Designer settings.

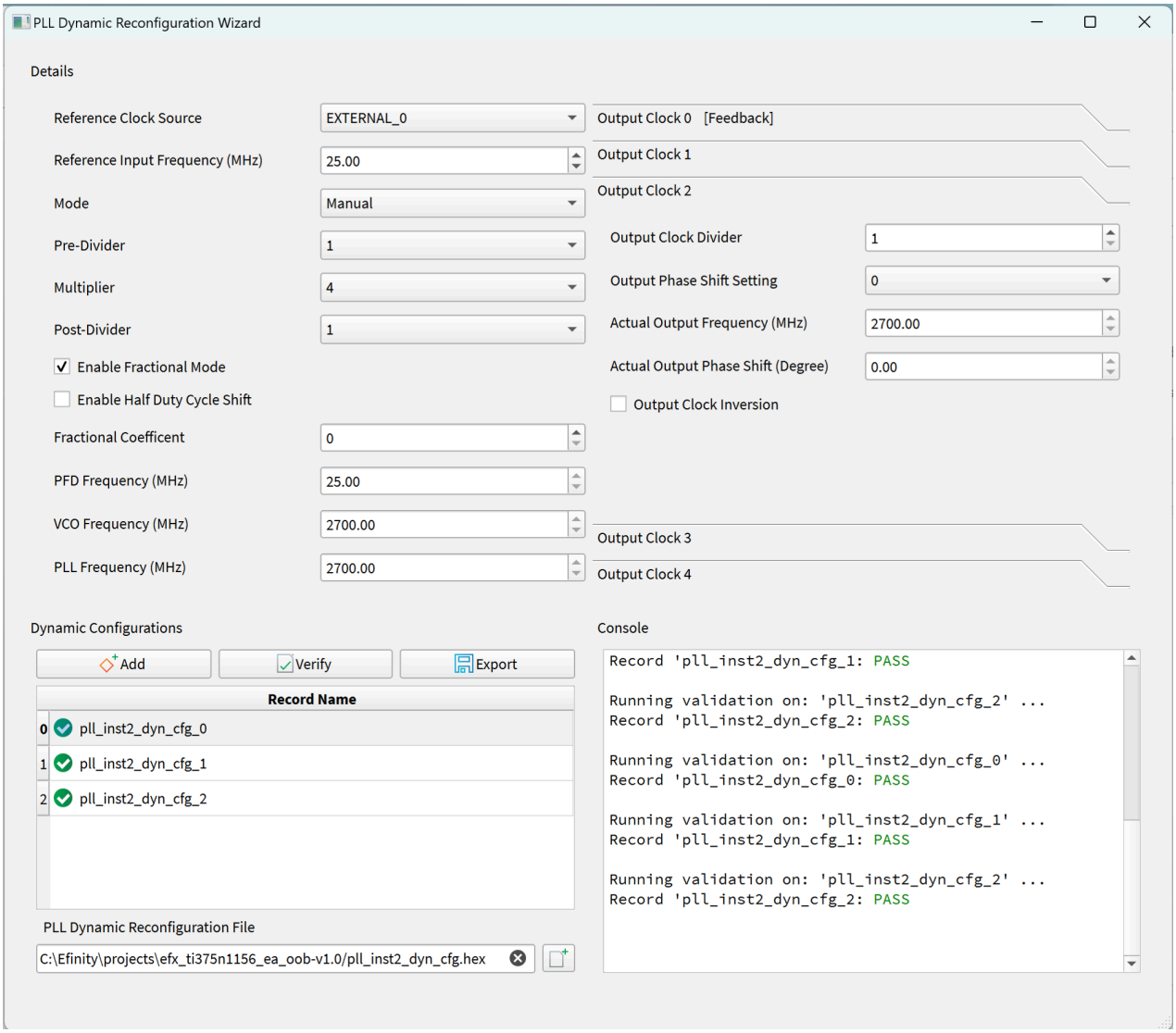
The PLL Dynamic Reconfiguration Core requires that you enable the PLL's reset and locked pins in the PLL Clock Calculator.

Table 92: Dynamic Reconfiguration Options

Parameter	Choices	Description
Enable	On, off	Turn on to enable dynamic reconfiguration.
Reference Clock Selector [1:0] Bus Name	User defined	Specify the bus name for the reference clock selector.
[2] Core Clock 0 Name	User defined	Specify the core clock name.
[3] Core Clock 1 Name	User defined	Specify the core clock name.
Configuration Clock Pin Name	User defined	Specify the configuration clock name.
Configuration Data Output Pin Name	User defined	Specify the configuration data output name.
Configuration Select Pin Name	User defined	Specify the configuration select name.
Reconfiguration Wizard	-	Click to open the wizard.

The PLL Dynamic Reconfiguration Wizard helps you define one or more PLL configuration sets that you use later to dynamically change the PLL's comfiguration.

Figure 52: PLL Dynamic Reconfiguration Wizard



To use the wizard:

1. Click **Add** to create a new configuration set.
2. Enter the PLL settings. Because the settings are complicated, Efinix suggests that you create a configuration first using the PLL Clock Calculator, note the resulting settings, and then enter them into the wizard.
3. Click **Verify**. The wizard checks your settings to make sure they are legal values. The results display in the **Console**.
4. Specify the output file name (**.hex**) in the **PLL Dynamic Reconfiguration File** box or use the default.
5. Click **Export** to save your configuration settings to the **.hex** file. Make note of the name; you specify this **.hex** file when you configure the PLL Dynamic Reconfiguration core.

Using the Fractional PLL Clock Calculator

The PLL Clock Calculator provides a graphical way for you to set up the advanced PLL block. When you open the calculator, the GUI appears in automatic mode, which provides basic settings. You can:

- Choose the feedback mode (**Local**, **Core**, or **External**).
- Turn signals on (gray x) or off (green arrow) by clicking the icons next to the signal.
- Specify the signal names.
- Specify the phase shift.
- Choose which clock has feedback (for core feedback mode).

The calculator has a manual mode. Refer to **Manually Configuring the PLL** on page 151 for details.

The Titanium PLL supports dynamic phase shifting. To enable it, click the **Dynamic** button for the clock output. The calculator adds three additional pins that you use to control the dynamic shifting. You need to specify the pin names.

As you make selections, the calculator determines the values for the pre-divider, multiplier, post divider, and clock dividers that meet your settings. The GUI prompts you if you make selections that are impossible to solve.

In manual mode, the interface displays the PLL's internal block diagram, and provides boxes for you to set the values for the pre-divider, multiplier, post divider, and clock dividers. As you adjust the values, the calculator prompts you if you make settings that result in F_{VCO} values that are out of range or are impossible to solve. When you turn manual mode off, the calculator adjusts the output clock frequencies to match the manual settings. If you have incorrect settings for the pre-divider, multiplier, post divider, and clock dividers, when you turn manual mode off, the calculator adjusts the values to ones that allow a valid solution.

When you are finished using the calculator, click **Finish** to save your settings and close the GUI.

Programmable Duty Cycle and Fractional Feedback

`CLKOUT1` supports programmable duty cycle and fractional modes; you cannot use both modes at the same time. To turn these options on:

1. Click the settings icon next to the Clock 1 options in the clock calculator.
2. In the **CLKOUT1 Feedback Mode** dialog box, choose the mode:
   - **Disable**—Do not use either mode (default).
   - **Programmable Duty Cycle**—When you choose this mode you can set the target duty cycle. The dialog box shows the actual duty cycle the PLL can achieve based on your other settings.
   - **Use as Fractional Feedback**—Choose this mode to allow the CLKOUT 1 clock to have a fractional part as well as the integer part. If you want to see the fractional

divider equations this mode enables, turn on Manual Mode and click the f icon in the **CLK Divider 0-4** boxes.

Design Check: Fractional PLL Messages

Note: The design rules in **Design Check: PLL Messages** on page 153 are also applicable to fractional PLL blocks. The messages in this topic apply to fractional PLLs only.

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

pll_rule_feedback_clock (error)

Message	Feedback clock name is required with non-internal feedback
To fix	Specify the clock pin name when using a non-internal feedback mode.

Message	Feedback clock name <string> is not from the same PLL
To fix	The clock name has to come from one of the output clock of the same PLL.

Message	Feedback clock in local mode has to connect to output clock 0 or 1
To fix	Set the local feedback mode to either output clock 0 or 1.

Message	Feedback clock <string> is not connected to pll clkout
To fix	Assign a feedback clock to one of the PLL output clock.

pll_rule_frac_en (error)

Message	Fractional mode enable status is invalid, <err_message>
To fix	Set Fractional mode enable status to valid value.

pll_rule_frac_feedback (error)

Message	Feedback mode should be local in fractional mode
To fix	Set feedback mode to local in the **PLL Clock Calculator**.

Message	Feedback clock is not configured in fractional mode
To fix	Add a feedback clock to Output Clock 1 in the **PLL Clock Calculator**

Message	Feedback clock must be Output Clock 1 in fractional mode
To fix	Set the feedback clock to Output Clock 1 in the **PLL Clock Calculator**.

pll_rule_frac_pdc (error)

Message	Fractional mode shouldn't be used together with programmable duty cycle
To fix	Disable either the programmable duty cycle or the fractional mode. You cannot use both modes at the same time.

pll_rule_frac_ssc (error)

Message	SSC feature requires PLL operates in fractional mode SSC feature requires PLL in local feedback mode SSC feature requires CLKOUT1 to be configured as feedback clock
To fix	To use SSC, the PLL must be in **Local** feedback mode with CLKOUT1 as the feedback clock, and fractional feedback must be turned on. See **Programmable Duty Cycle and Fractional Feedback** on page 164 for steps to enable fractional feedback mode.

pll_rule_ssc_mode (error)

Message	SSC Mode is invalid, <err_message>
To fix	Set the SSC Mode to either DISABLE, STATIC, or DYNAMIC.

pll_rule_ssc_freq (error)

Message	SSC Frequency is invalid, <err_message>
To fix	Update SSC frequency to a valid range.

pll_rule_ssc_amp (error)

Message	SSC Amplitude is invalid, <err_message>
To fix	Update SSC Amplitude to a valid range.

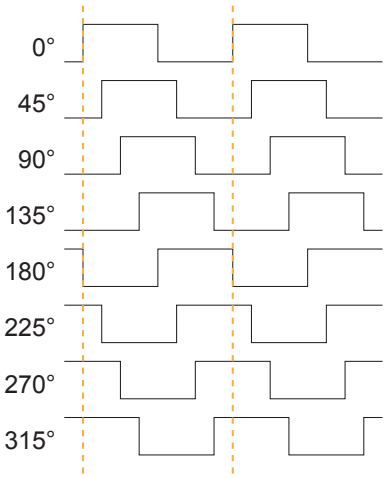
Understanding PLL Phase Shifting

The PLL supports clock phases from 0 to 315 degrees.

- You can set a phase shift for a clock output in the PLL Clock Calculator. The calculator tries to get as close to that number as it can.
- If you want to set the phase shift manually, where you use the options to set the M, N, O, divider, and shift values, the phase shift results you get are 0, 45, 90, 135, 180, 225, 270, and 315.

You can also control the phase shifting dynamically. To turn on dynamic phase shifting, click the Dynamic button next to the clock frequency in the PLL Clock Calculator. The GUI adds three pins to control the shift. the shift select, and the shift enable. You can dynamically change the phases simultaneously.

Figure 53: Example PLL Clock Phases



About the Spread-Spectrum Clocking PLL Interface

The Titanium MIPI D-PHY interface includes a spread-spectrum clocking (SSC) PLL that spreads or varies the signal spectrum around the ideal clock frequency. If you are not using the MIPI D-PHY TX interface for MIPI signals, you can use the SSC PLL as another clock source.

Table 93: SSC PLL Support

Family	Available In
Titanium	Ti85, Ti135, Ti90, Ti120, Ti165, Ti180, Ti240, and Ti375

The PLL consists of a pre-divider counter (N counter), a feedback multiplier counter (M counter), a post-divider counter (O counter), and output divider (C). You cannot modify the counter settings. Instead, you specify the output frequency you want and the reference clock frequency. If the SSC PLL cannot exactly match the output frequency, it displays (and uses) the frequency that is closest to your setting.

By default, the SSC PLL acts as a regular PLL. You enable the spread-spectrum clocking by turning on the **Enable Spread Spectrum Clock (SSC)** option in the Interface Designer.

Figure 54: SSC PLL Block Diagram

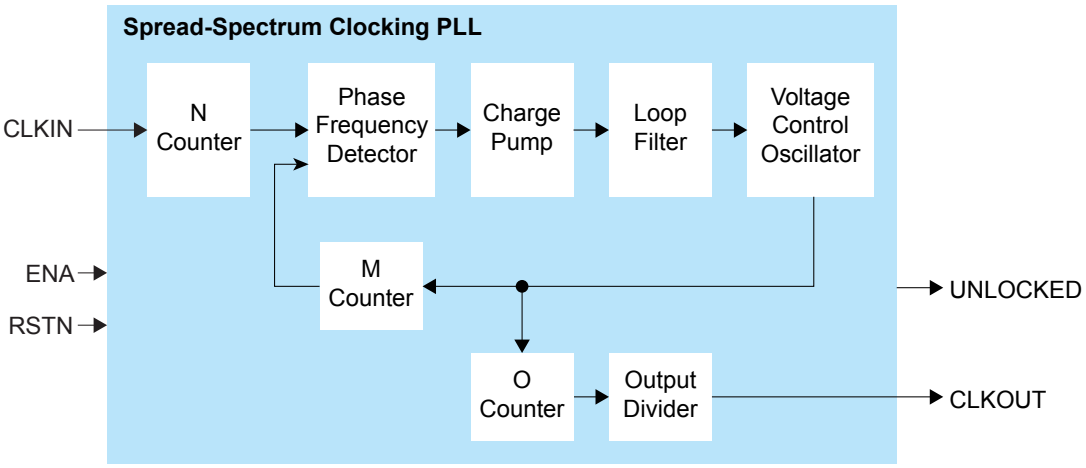


Figure 55: SSC PLL Interface Block Diagram

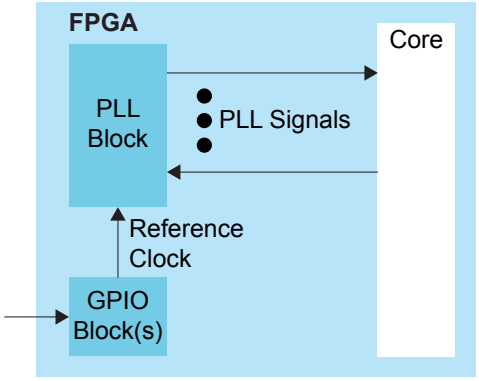


Table 94: SSC PLL Signals (Interface to FPGA Fabric)

Signal	Direction	Description
CLKIN	Input	Reference clocks from core, PLL, or GPIO.
CLKOUT	Output	PLL SSC Clock Out Pin Name.
RSTN	Input	Active-low PLL SSC reset signal.
UNLOCKED	Output	PLL Unlock State Pin Name. Goes high when PLL SSC is in unlock state. Connect this signal in your design to monitor the lock status.
ENA	Input	(Optional) PLL SSC Enable Pin Name: Always enable: 1 Disable: 0 Can be driven by an active signal for dynamic enable.

Table 95: SSC PLL Interface Designer Settings (Base Tab)

Parameter	Choices	Notes
Clockout Frequency (MHz)	5 - 312.5	Click **Update Output Clock Frequency** and select the valid frequency.
Instance Name	User defined	
PLL SSC Resource	None, MIPI_TX0, MIPI_TX1, MIPI_TX2, MIPI_TX3	Choose the resource.
Reference Clock Frequency	12, 19.2, 25, 26, 27, 38.4, 52	Choose reference clock in MHz.
Reference Clock Source Type	pll, gpio, core	Choose which resource generates the reference clock. For **gpio** and **pll**, the Block Editor shows you which resource to connect as the reference clock. For **core**, you specify the clock name.

Table 96: SSC PLL Interface Designer Settings (Control/Status Tab)

Option	Choices	Notes
Enable Spread Spectrum Clock (SSC)	On or off	Turn on to enable SSC. Turn off to use the SSC PLL as a regular PLL without the waveform variation.
SSC Frequency (kHz)	30 - 33	Spread-spectrum clock frequency setting. Default: 30
SSC Initial Amplitude (PPM)	2501 - 5000	Spread-spectrum clock initial spread down amount in ppm. The initial amplitude value must be larger than the amplitude value. Default: 5000
SSC Amplitude (PPM)	2500 - 4999	Spread-spectrum clock amount in ppm. Default: 4999
Clockout Connection Type	gclk, rclk	Choose whether to connect to a global clock (**gclk**) or regional clock (**rclk**). Default: gclk
<*description*> Pin Name	User defined	Control and status pin names. Efinix recommends that you use the defaults.

Using the SSC PLL Block

The SSC PLL block lets you configure the spread-spectrum clocking PLL.

In the **Base** tab, specify the clock output frequency, instance name, and choose the resource.

Important: The SSC PLL block uses a MIPI D-PHY TX resource. You cannot use the same MIPI TX resource for a MIPI DPHY TX block and the SSC PLL. (You use the resource for one or the other.)

Reference Clock Settings

The SSC PLL reference clock can come from the FPGA core, a GPIO, or from a PLL clock output. You select the clocks using the **Reference Clock Source Type** drop-down box:

- **core**—The reference clock comes from the FPGA core. Specify the clock signal name in the **Reference Clock** box.

- **gpio—**The PLL reference clock comes from an external pin. The GUI display which GPIO resource to use.

  Note: In this mode, a GPIO block with a **mipi_clkin** connection type must generate the reference clock. The software displays which resource you can use (and the instance name if you have created it).

  1. Add a GPIO block.
  2. Enter the instance name.
  3. Choose **input** as the mode.
  4. In the **Input** tab, choose **Connection Type > mipi_clkin**.
  5. In the Resource Assigner, assign the GPIO to the resource shown in the SSC PLL's **Base** tab.
- **pll—**The reference clock comes from a PLL clock output. The GUI shows which PLL to use. See Using the PLL Block on page 150 for steps to create the PLL block.

Control / Status Settings

If you are using the spread-spectrum feature of the PLL, turn on the Enable Spread Spectrum Clock (SSC) option. Then, set the frequency and amplitude for the spread spectrum clock.

Choose whether the SSC PLL's output clock connects to a GCLK or RCLK.

Optionally, you can change the default pin names for the block.

Design Check: PLL SSC Errors

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

pll_ssc_rule_inst_name (error)

Message	Instance name is empty
To fix	Enter the instance name.

pll_ssc_rule_ref_clk_resource (error)

Message	Reference clock resource <res_name> is not configured as mipi_clkin connection
To fix	You get this error when the reference clock resource (GPIO block) is set to **gpio** instead of **mipi_clkin**. Change the setting for the GPIO block.

Message	Reference clock resource <res_name> input name is empty
To fix	Specify the name.

Message	Reference clock resource <res_name> is not configured as input
To fix	The GPIO block you are using as the reference clock must be configured as an input. Change the **Mode** setting for the GPIO block.

Message	Reference clock GPIO resource <res_name> has not been configured
To fix	You added the GPIO block but did not configure it correctly. It should be an input and should be configured as **Connection Type** > **mipi_clkin**.

Message	Reference clock PLL resource <res_name> Output Clock <#> has not been configured
To fix	If you choose **Reference Clock Source Type** > **pll**, you also need to configure the output clock for the PLL resource shown in the **Base** tab.

Message	Reference clock PLL resource <res_name> has not been configured
To fix	If you choose **Reference Clock Source Type** > **pll**, you also need to configure the PLL resource shown in the **Base** tab.

Message	Core reference clock pin name is empty
To fix	If you choose **Reference Clock Source Type** > **core**, then you need to enter the name for the pin.

pll_ssc_rule_pll_outclk (error)

Message	Empty PLL clock out pin name
To fix	You need to specify the output clock pin name.

pll_ssc_rule_auto_cal_freq (warning)

Message	Output Clock frequency is not match with actual one (Expected: <User Input>, Actual: <The freq that PLL can get>)
To fix	Click the **Update Output Clock Frequency** button to select the closest frequency that the PLL can get.

pll_ssc_rule_ssc_amp (error)

Message	SSC Amplitude must be less than SSC Init Amplitude
To fix	Change the values so that the SSC Amplitude is less than SSC Init Amplitude.

Chapter 12

Oscillator

Contents:

The Titanium FPGA has 1 low-frequency oscillator tailored for low-power operation. The oscillator runs at a nominal frequency of 10, 20, 40, or 80 MHz. You can use the oscillator to perform always-on functions with the lowest power possible. It's output clock is available to the core. You can enable or disable the oscillator to allow power savings when not in use. The oscillator has:

- An output duty cycle of 45% to 55%.
- A ±20% frequency variation from device to device.

Using the Oscillator Block

To use the oscillator block in your design:

1. Add the oscillator block.
2. Select the resource (**OSC_0**).
3. Choose the clock frequency (**10**, **20**, **40**, or **80**).
4. Specify the instance name and clock pin.

Note: You can disable the internal oscillator in Titanium FPGAs. The internal oscillator is disabled if it is not instantiated in the Efinity® Interface Designer.

Design Check: Oscillator Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

osc_rule_clock_name (error)

Message	Clock pin name is not specified Valid characters are alphanumeric characters with dash and underscore only
To fix	Specify a valid clock name.

osc_rule_ena_pin_transceiver (error)

Message	The oscillator clock cannot be tri-state and has to be free running for *<transceiver type>* usage
To fix	Remove the enable pin name.

osc_rule_frequency (error)

Message	Invalid frequency <value> Unrecognized frequency <value>
To fix	The oscillator only supports 10, 20, 40, or 80 MHz frequencies, so choose one of those.

osc_rule_instance_count (error)

Message	There can only be one oscillator instance
To fix	Titanium FPGAs only have one oscillator, so you can only use one oscillator block.

osc_rule_inst_name (error)

Message	Instance name is empty Valid characters are alphanumeric characters with dash and underscore only
To fix	Specify a valid instance name.

osc_rule_resource (error)

Message	Resource name is empty Resource is not a valid oscillator device instance
To fix	Although Titanium FPGAs only have one oscillator, you still need to choose the resource when you create an oscillator block.

Chapter 13

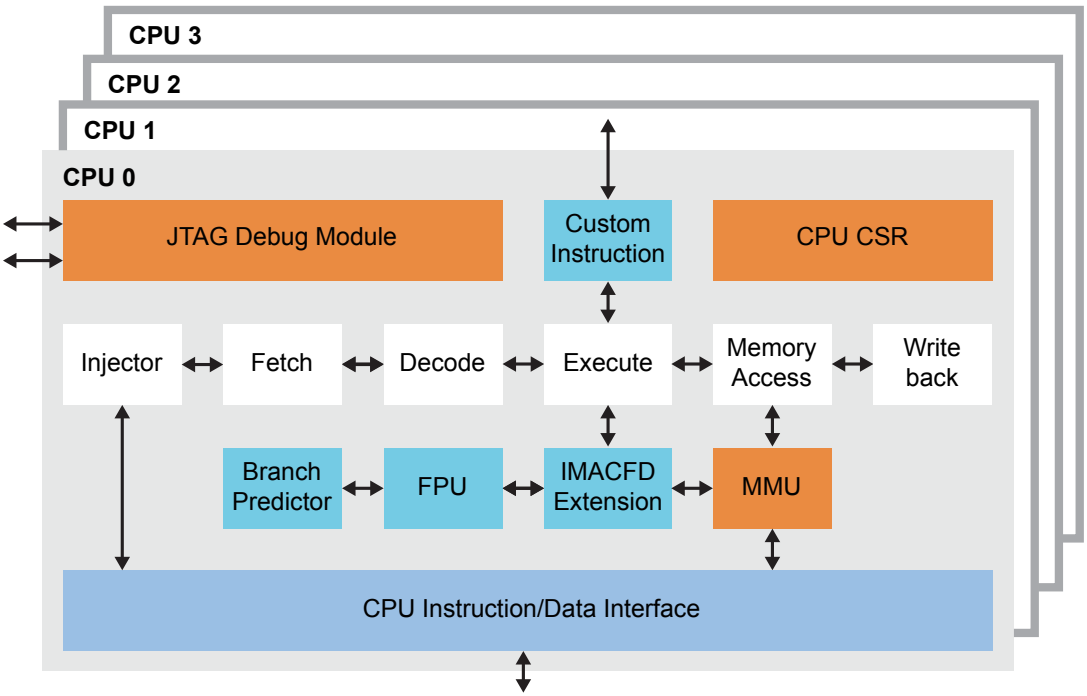
Hardened RISC-V Block Interface

Contents:

Important: All information is preliminary and pending definition.

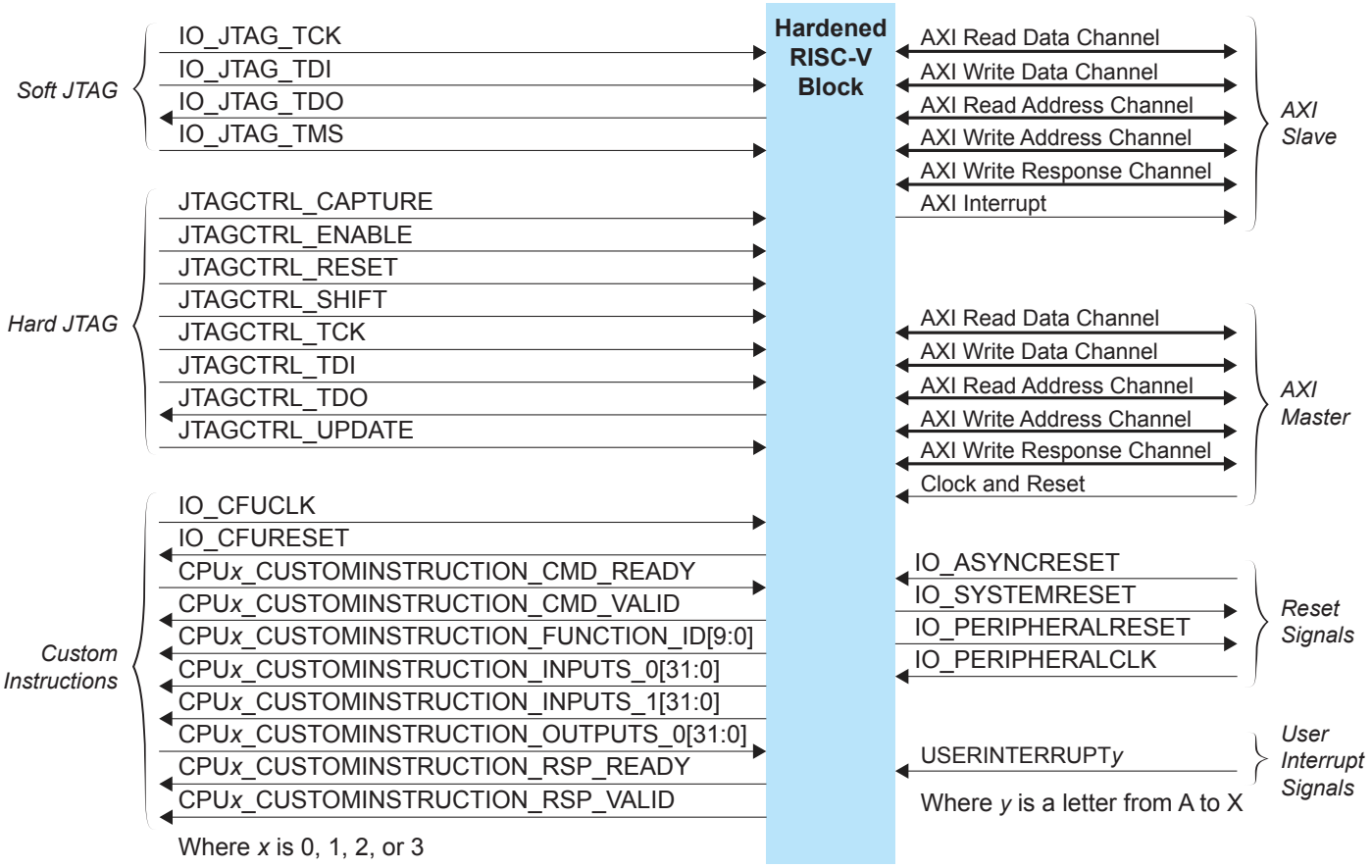
Titanium FPGAs have a hardened RISC-V block with a 32-bit CPU featuring the ISA RISCV32I with M, A, C, F, and D extensions, and six pipeline stages (fetch, injector, decode, execute, memory, and writeback). The hard processor has 4 CPUs each with a dedicated FPU and custom instructions. The processor supports the standard RISC-V debug specification with 8 hardware breakpoints as well as machine and supervisor privileged mode, and Linux MMU SV32 page-based virtual memory.

Figure 56: Hardened RISC-V Block Overview



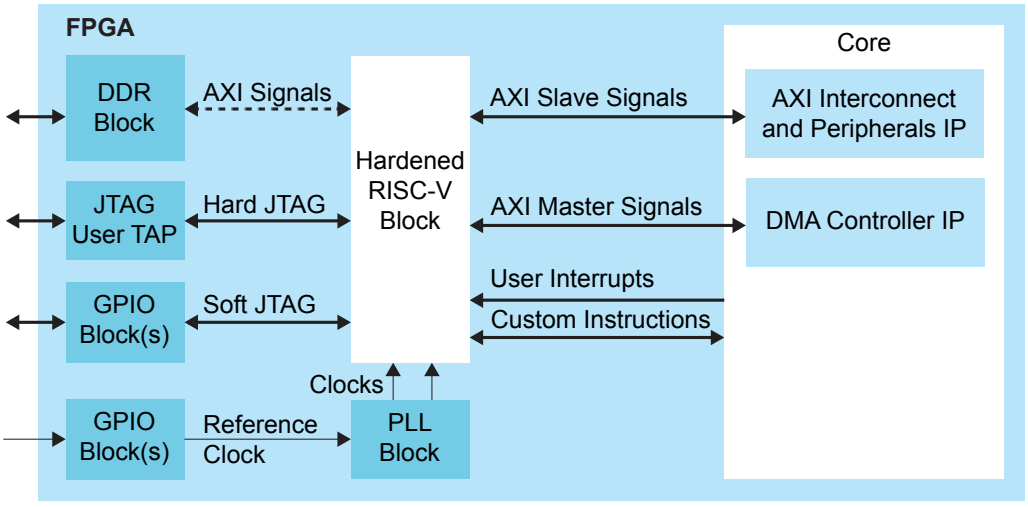
This topic provides an overview of the hardened RISC-V block and the signals that connect to the Titanium's core fabric and interfaces. For complete details on the processor and it's specifications, refer to the Sapphire High-Performance SoC Data Sheet.

Figure 57: Hardened RISC-V Block Diagram



The hardened RISC-V block is connected directly to port 1 on the hard LPDDR4 controller; therefore, you do not need to implement those connections. Your design should implement the other interface blocks as needed.

Figure 58: Hardened RISC-V Block Interface Block Diagram



The PLLs that can feed the RISC-V system clock are BL0 CLKOUT1, BL1 CLKOUT2, or BL2 CLKOUT1.

The PLLs that can feed the RISC-V memory clock are BL0 CLKOUT2, BL1 CLKOUT1, or BL2 CLKOUT2.

Note: The PLL that clocks the hardened RISC-V block should not use fractional output or spread-spectrum clocking because these features increase jitter.

AXI4 Slave Interface for Peripherals

The AXI slave interface connects to user-defined peripherals through an AXI interconnect bus. You use the IP Manager to build the AXI interconnect and peripherals.

Note: See the Titanium Interfaces User Guide for more details.

Table 97: AXI Interrupt

Port	Direction	Clock Domain
AXIAINTERRUPT	Output	io_peripheralClk

Table 98: AXI Slave Read Address Channel

Port	Direction	Clock Domain	Description
AXIA_ARADDR[31:0]	Input	io_peripheralClk	Read address. It gives the address of the first transfer in a burst transaction.
AXIA_ARBURST[1:0]	Input	io_peripheralClk	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated. 'b01 = INCR 'b10 = WRAP
AXIA_ARCACHE[3:0]	Input	io_peripheralClk	Memory type. This signal indicates how transactions are required to progress through a system.
AXIA_ARLEN[7:0]	Input	io_peripheralClk	Burst length. This signal indicates the number of transfers in a burst.
AXIA_ARLOCK	Input	io_peripheralClk	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
AXIA_ARPROT[2:0]	Input	io_peripheralClk	Defines the access permissions for read accesses.
AXIA_ARQOS[3:0]	Input	io_peripheralClk	QoS identifier for read transaction.
AXIA_ARREADY	Output	io_peripheralClk	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
AXIA_ARREGION[3:0]	Input	io_peripheralClk	Region identifier. Permits a single physical interface on a slave to be used for multiple logical interfaces.
AXIA_ARSIZE[2:0]	Input	io_peripheralClk	Burst size. This signal indicates the size of each transfer in the burst.
AXIA_ARVALID	Input	io_peripheralClk	Address valid. This signal indicates that the channel is signaling valid address and control information.

Table 99: AXI Slave Write Address Channel

Port	Direction	Clock Domain	Description
AXIA_AWADDR[31:0]	Input	io_peripheralClk	Write address. It gives the address of the first transfer in a burst transaction.
AXIA_AWBURST[1:0]	Input	io_peripheralClk	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated.
AXIA_AWCACHE[3:0]	Input	io_peripheralClk	Memory type. This signal indicates how transactions are required to progress through a system.
AXIA_AWLEN[7:0]	Input	io_peripheralClk	Burst length. This signal indicates the number of transfers in a burst.
AXIA_AWLOCK	Input	io_peripheralClk	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
AXIA_AWPROT[2:0]	Input	io_peripheralClk	Defines the access permissions for write accesses.
AXIA_AWQOS[3:0]	Input	io_peripheralClk	QoS identifier for write transaction.
AXIA_AWREADY	Output	io_peripheralClk	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
AXIA_AWREGION[3:0]	Input	io_peripheralClk	Region identifier. Permits a single physical interface on a slave to be used for multiple logical interfaces.
AXIA_AWSIZE[2:0]	Input	io_peripheralClk	Burst size. This signal indicates the size of each transfer in the burst.
AXIA_AWVALID	Input	io_peripheralClk	Address valid. This signal indicates that the channel is signaling valid address and control information.

Table 100: AXI Slave Write Response Channel

Port	Direction	Clock Domain	Description
AXIA_BREADY	Input	io_peripheralClk	Response ready. This signal indicates that the master can accept a write response.
AXIA_BRESP[1:0]	Output	io_peripheralClk	Read response. This signal indicates the status of the read transfer.
AXIA_BVALID	Output	io_peripheralClk	Write response valid. This signal indicates that the channel is signaling a valid write response.

Table 101: AXI Slave Read Data Channel

Port	Direction	Clock Domain	Description
AXIA_RDATA[31:0]	Output	io_peripheralClk	Read data.
AXIA_RLAST	Output	io_peripheralClk	Read last. This signal indicates the last transfer in a read burst.
AXIA_RREADY	Input	io_peripheralClk	Read ready. This signal indicates that the master can accept the read data and response information.
AXIA_RRESP[1:0]	Output	io_peripheralClk	Read response. This signal indicates the status of the read transfer.
AXIA_RVALID	Output	io_peripheralClk	Read valid. This signal indicates that the channel is signaling the required read data.

Table 102: AXI Slave Write Data Channel

Port	Direction	Clock Domain	Description
AXIA_WDATA[31:0]	Input	io_peripheralClk	Write data.
AXIA_WLAST	Input	io_peripheralClk	Write last. This signal indicates the last transfer in a write burst.
AXIA_WREADY	Output	io_peripheralClk	Write ready. This signal indicates that the slave can accept the write data.
AXIA_WSTRB[3:0]	Input	io_peripheralClk	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
AXIA_WVALID	Input	io_peripheralClk	Write valid. This signal indicates that valid write data and strobes are available.

AXI Interface to DMA

This AXI master interface has a 128-bit data channel to connect to the DMA controller IP core.

Table 103: Clock and Reset

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0_CLK	Input		
IO_DDRMASTERS_0_RESET	Input	IO_DDRMASTERS_0_CLK	

Table 104: AXI Master Read Address Channel

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0_ AR_PAYLOAD_ADDR[31:0]	Output	IO_DDRMASTERS_0_CLK	Read address. It gives the address of the first transfer in a burst transaction.
IO_DDRMASTERS_0 _AR_PAYLOAD_BURST[1:0]	Output	IO_DDRMASTERS_0_CLK	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated. 'b01 = INCR 'b10 = WRAP
IO_DDRMASTERS_0 _AR_PAYLOAD_CACHE[3:0]	Output	IO_DDRMASTERS_0_CLK	Memory type. This signal indicates how transactions are required to progress through a system.
IO_DDRMASTERS_0 _AR_PAYLOAD_ID[3:0]	Output	IO_DDRMASTERS_0_CLK	Address ID. This signal identifies the group of address signals.
IO_DDRMASTERS_0 _AR_PAYLOAD_LEN[7:0]	Output	IO_DDRMASTERS_0_CLK	Burst length. This signal indicates the number of transfers in a burst.
IO_DDRMASTERS_0 _AR_PAYLOAD_LOCK	Output	IO_DDRMASTERS_0_CLK	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
IO_DDRMASTERS_0 _AR_PAYLOAD_PROT[2:0]	Output	IO_DDRMASTERS_0_CLK	Defines the access permissions for read accesses.
IO_DDRMASTERS_0 _AR_PAYLOAD_QOS[3:0]	Output	IO_DDRMASTERS_0_CLK	QoS identifier for read transaction.

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _AR_PAYLOAD_REGION[3:0]	Output	IO_DDRMASTERS_0_CLK	Region identifier. Permits a single physical interface to be used for multiple logical interfaces.
IO_DDRMASTERS_0 _AR_PAYLOAD_SIZE[2:0]	Output	IO_DDRMASTERS_0_CLK	Burst size. This signal indicates the size of each transfer in the burst.
IO_DDRMASTERS_0 _AR_READY	Input	IO_DDRMASTERS_0_CLK	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
IO_DDRMASTERS_0_AR_VALID	Output	IO_DDRMASTERS_0_CLK	Address valid. This signal indicates that the channel is signaling valid address and control information.

Table 105: AXI Master Write Address Channel

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _AW_VALID	Output	IO_DDRMASTERS_0_CLK	Address valid. This signal indicates that the channel is signaling valid address and control information.
IO_DDRMASTERS_0 _AW_READY	Input	IO_DDRMASTERS_0_CLK	Address ready. This signal indicates that the slave is ready to accept an address and associated control signals.
IO_DDRMASTERS_0 _AW_PAYLOAD_ADDR[31:0]	Output	IO_DDRMASTERS_0_CLK	Write address. It gives the address of the first transfer in a burst transaction.
IO_DDRMASTERS_0 _AW_PAYLOAD_ID[3:0]	Output	IO_DDRMASTERS_0_CLK	Address ID. This signal identifies the group of address signals.
IO_DDRMASTERS_0 _AW_PAYLOAD_REGION[3:0]	Output	IO_DDRMASTERS_0_CLK	Region identifier. Permits a single physical interface to be used for multiple logical interfaces.
IO_DDRMASTERS_0 _AW_PAYLOAD_LEN[7:0]	Output	IO_DDRMASTERS_0_CLK	Burst length. This signal indicates the number of transfers in a burst.
IO_DDRMASTERS_0 _AW_PAYLOAD_SIZE[2:0]	Output	IO_DDRMASTERS_0_CLK	Burst size. This signal indicates the size of each transfer in the burst.
IO_DDRMASTERS_0 _AW_PAYLOAD_BURST[1:0]	Output	IO_DDRMASTERS_0_CLK	Burst type. The burst type and the size determine how the address for each transfer within the burst is calculated.
IO_DDRMASTERS_0 _AW_PAYLOAD_LOCK	Output	IO_DDRMASTERS_0_CLK	Lock type. This signal provides additional information about the atomic characteristics of the transfer.
IO_DDRMASTERS_0 _AW_PAYLOAD_CACHE[3:0]	Output	IO_DDRMASTERS_0_CLK	Memory type. This signal indicates how transactions are required to progress through a system.
IO_DDRMASTERS_0 _AW_PAYLOAD_QOS[3:0]	Output	IO_DDRMASTERS_0_CLK	QoS identifier for write transaction.
IO_DDRMASTERS_0 _AW_PAYLOAD_PROT[2:0]	Output	IO_DDRMASTERS_0_CLK	Defines the access permissions for write accesses.

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _AW_PAYLOAD_ALLSTRB	Output	IO_DDRMASTERS_0_CLK	Write all strobes asserted. The DDR controller only supports a maximum of 16 AXI beats for write commands using this signal.

Table 106: AXI Master Write Response Channel

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _B_PAYLOAD_ID[3:0]	Input	IO_DDRMASTERS_0_CLK	Response ID tag. This signal is the ID tag of the write response.
IO_DDRMASTERS_0 _B_PAYLOAD_RESP[1:0]	Input	IO_DDRMASTERS_0_CLK	Read response. This signal indicates the status of the read transfer.
IO_DDRMASTERS_0_B_READY	Output	IO_DDRMASTERS_0_CLK	Response ready. This signal indicates that the master can accept a write response.
IO_DDRMASTERS_0_B_VALID	Input	IO_DDRMASTERS_0_CLK	Write response valid. This signal indicates that the channel is signaling a valid write response.

Table 107: AXI Master Read Data Channel

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _R_PAYLOAD_DATA[127:0]	Input	IO_DDRMASTERS_0_CLK	Read data.
IO_DDRMASTERS_0 _R_PAYLOAD_ID[3:0]	Input	IO_DDRMASTERS_0_CLK	Read ID tag. This signal is the identification tag for the read data group of signals generated by the slave.
IO_DDRMASTERS_0 _R_PAYLOAD_LAST	Input	IO_DDRMASTERS_0_CLK	Read last. This signal indicates the last transfer in a read burst.
IO_DDRMASTERS_0 _R_PAYLOAD_RESP[1:0]	Input	IO_DDRMASTERS_0_CLK	Read response. This signal indicates the status of the read transfer.
IO_DDRMASTERS_0_R_READY	Output	IO_DDRMASTERS_0_CLK	Read ready. This signal indicates that the master can accept the read data and response information.
IO_DDRMASTERS_0_R_VALID	Input	IO_DDRMASTERS_0_CLK	Read valid. This signal indicates that the channel is signaling the required read data.

Table 108: AXI Master Write Data Channel

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0_W_VALID	Output	IO_DDRMASTERS_0_CLK	Write valid. This signal indicates that valid write data and strobes are available.
IO_DDRMASTERS_0 _W_READY	Input	IO_DDRMASTERS_0_CLK	Write ready. This signal indicates that the slave can accept the write data.
IO_DDRMASTERS_0 _W_PAYLOAD_DATA[127:0]	Output	IO_DDRMASTERS_0_CLK	Write data.

Port	Direction	Clock Domain	Description
IO_DDRMASTERS_0 _W_PAYLOAD_LAST	Output	IO_DDRMASTERS_0_CLK	Write last. This signal indicates the last transfer in a write burst.
IO_DDRMASTERS_0 _W_PAYLOAD_STRB[15:0]	Output	IO_DDRMASTERS_0_CLK	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.

JTAG Signals

The hardened RISC-V block includes two sets of JTAG signals. The soft JTAG connects to I/O blocks while the hard JTAG connects to the JTAG User TAP interface block.

Table 109: Soft JTAG Ports

Port	Direction	Clock Domain	
IO_JTAG_TCK	Input	IO_JTAG_TCK	JTAG test clock pin.
IO_JTAG_TDI	Input	IO_JTAG_TCK	JTAG test data in pin.
IO_JTAG_TDO	Output	IO_JTAG_TCK	JTAG test data out pin.
IO_JTAG_TMS	Input	IO_JTAG_TCK	JTAG mode select pin.

Table 110: Hard JTAG Ports

Port	Direction	Clock Domain	Description
JTAGCTRL_CAPTURE	Input	JTAGCTRL_TCK	Capture pin.
JTAGCTRL_ENABLE	Input	JTAGCTRL_TCK	Enable the JTAG user TAP interface.
JTAGCTRL_RESET	Input	JTAGCTRL_TCK	Reset.
JTAGCTRL_SHIFT	Input	JTAGCTRL_TCK	Shift pin.
JTAGCTRL_TCK	Input	JTAGCTRL_TCK	JTAG test clock pin.
JTAGCTRL_TDI	Input	JTAGCTRL_TCK	JTAG test data in pin.
JTAGCTRL_TDO	Output	JTAGCTRL_TCK	JTAG test data out pin.
JTAGCTRL_UPDATE	Input	JTAGCTRL_TCK	Update pin.

Custom Instruction Signals

The hardened RISC-V interface has two 32-bit custom instruction interfaces for each CPU. The custom instructions use a type R opcode.

Table 111: Custom Instructions

Where *n* is 0, 1, 2, or 3 for the CPU number.

Port	Direction	Clock Domain	Description
IO_CFUCLK	Input		Clock.
IO_CFURESET	Output	IO_CFUCLK	Reset.
CPU*n*_CUSTOMINSTRUCTION _CMD_READY	Input	IO_CFUCLK	Indicates that the custom processing logic is ready to process register rs1 and rs2 from the CPU.

Port	Direction	Clock Domain	Description
CPU*n*_CUSTOMINSTRUCTION_CMD_VALID	Output	IO_CFUCLK	Indicates that registers rs1 and rs2 are present and ready for processing.
CPU*n*_CUSTOMINSTRUCTION_FUNCTION_ID[9:0]	Output	IO_CFUCLK	Function id for the custom instruction.
CPU*n*_CUSTOMINSTRUCTION_INPUTS_0[31:0]	Output	IO_CFUCLK	Register rs1 for the custom instruction.
CPU*n*_CUSTOMINSTRUCTION_INPUTS_1[31:0]	Output	IO_CFUCLK	Register rs2 for the custom instruction.
CPU*n*_CUSTOMINSTRUCTION_OUTPUTS_0[31:0]	Input	IO_CFUCLK	Result of the custom instruction.
CPU*n*_CUSTOMINSTRUCTION_RSP_READY	Output	IO_CFUCLK	Indicates that the CPU is ready to accept the custom instruction result.
CPU*n*_CUSTOMINSTRUCTION_RSP_VALID	Output	IO_CFUCLK	Indicates that the custom instruction result is available.

User Interrupt Signals

Table 112: User Interrupts

Where *n* is a letter A-X.

Port	Direction	Description
USERINTERRUPT*n*	Input	Interrupt signal for a peripheral.

Clock Signals

Table 113: Clock

Port	Direction	Description
IO_PERIPHERALCLK	Input	Provides a clock for the peripherals and AXI slave interface.

Reset Signals

Table 114: Reset

Port	Direction	Description
IO_ASYNCRESET	Input	Active-high asynchronous reset for the entire system.
IO_SYSTEMRESET	Output	Synchronous active-high reset for the system clock.
IO_PERIPHERALRESET	Output	Synchronous active-high reset for the peripheral clock (io_peripheralClock).

Using the Hardened RISC-V Block

To use the Titanium hardened RISC-V block you add it to your interface design and configure the settings.

Instead of manually adding a RISC-V block to your interface design, Efinix recommends that you use the IP Manager to create a Sapphire High-Performance SoC instance. The IP Manager automatically creates all the Interface Designer blocks that you need for the SoC, so you do not have to add them manually.

Important: If you use the IP Manager to generate the Interface Designer blocks, **do not** change the settings of those blocks later in the Interface Designer. Otherwise, you will break your design.

Exception: You can specify a configuration file for on-chip RAM in the **Quad-Core RISC-V** > **Block Editor** > **Base tab**.

Table 115: Base Tab

Parameter	Choices	Description
Instance Name	User defined	Enter the instance name.
SOC Resource	SOC_0	Choose the resource. The IP Manager chooses SOC_0 by default.
On-Chip Ram Configuration File	User specified	Indicate the file for the on-chip memory configuration. You can specify the filename even if you auto-generated the RISC-V block with the IP Manager.

Table 116: Clock/Control Tab

Parameter	Choices	Description
System Clock Source	None, Clock 0, Clock 1, Clock 2	Choose the clock source. The IP Manager chooses the resource PLL_BL0 and Clock 0 for the system clock.
Memory Clock Source	None, Clock 0, Clock 1, Clock 2	Choose the clock source. The IP Manager chooses the resource PLL_BL1 and Clock 1 for the memory clock.
Active-High Periphery Controller Reset Pin Name	User defined	Efinix recommends you use the default pin names.
Active-High System Reset Pin Name		
Active-High asynchronous reset for SOC Pin Name		
Periphery Controller Clock Pin Name		
Invert Periphery Controller Clock Pin	On, off (default)	Turn on to invert the clock.

Table 117: User AXI Master Tab

Parameter	Choices	Description
Enable AXI Master Interface	On, off	Turn on to enable the interface. If you turn on the AXI Master in the IP Manager, the IP Manager enables the interface.
User AXI Master Clock Pin Name	User defined	Enter the pin name. If you turn on the AXI Master in the IP Manager, the IP Manager enters the default pin name.
Invert User AXI Master Clock Pin	On, off	Turn on to invert the clock.
User AXI Master Reset Pin Name	User defined	Enter the pin name. If you turn on the AXI Master in the IP Manager, the IP Manager enters the default pin name.
Read Address Channel tab Write Address Channel tab Write Response Channel tab Read Data Channel tab Write Data Channel tab	User defined	The Interface Designer shows default names for the AXI pins. Efinix recommends you keep the default values.

Table 118: User AXI Slave Tab

Parameter	Choices	Description
User AXI Slave Channel Interrupt Pin Name	User defined	Enter the pin name. The IP Manager enters the default pin name.
Read Address Channel tab Write Address Channel tab Write Response Channel tab Read Data Channel tab Write Data Channel tab	User defined	The Interface Designer shows default names for the AXI pins. Efinix recommends you keep the default values.

Table 119: Custom Instruction Tab

Where *n* is the interface number (0, 1, 2, or 3)

Parameter	Choices	Description
Eable Custom Instruction Interface *n*	On, off	Turn on to enable the interface.
Active Synchronous Reset for Custom Instruction Unit Pin Name	User defined	Enter the pin name.
Custom Instruction Unit Clock Pin Name	User defined	Enter the pin name.
Invert Custom Instruction Unit Clock Pin Name	On, off	Turn on to invert the pin.
Interface *n* tab	User defined	The Interface Designer shows default names for the custom instruction pins. Efinix recommends you keep the default values.

Table 120: External Interrupt Tab

Where *x* is a letter (A-X)

Parameter	Choices	Description
External Interrupt *x*: Pin Name	User defined	Enter the pin name.

Table 121: Debug Tab

Parameter	Choices	Description
JTAG Interface Type	FPGA, CPU, DISABLE	DISABLE: Do not use the JTAG interface. FPGA: Connect the interface to the JTAG User Tap. CPU: Connect the interface to GPIO pins.
Pin Names	User defined	Enter the pin names.

Design Check: RISC-V Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

qcrv32_rule_axi_master_empty_pins (error)

Message	Empty pin names found: <Pin description names>
To fix	You need to specify the pin names.

qcrv32_rule_axi_master_invalid_pins (error)

Message	Invalid pin names found: <pin description names>
To fix	Valid characters are alphanumeric characters with dash and underscore only.

qcrv32_rule_base_empty_pins (error)

Message	Empty pin names found: <Pin description names>
To fix	You need to specify the pin names.

qcrv32_rule_base_invalid_pins (error)

Message	Invalid pin names found: <pin description names>
To fix	Valid characters are alphanumeric characters with dash and underscore only.

qcrv32_rule_custom_instr_empty_pins (error)

Message	Empty pin names found: <Pin description names>
To fix	You need to specify the pin names.

qcrv32_rule_custom_instr_invalid_pins (error)

Message	Invalid pin names found: <pin description names>
To fix	Valid characters are alphanumeric characters with dash and underscore only.

qcrv32_rule_inst_name (error)

Message	Instance name is empty.
To fix	You need to specify the pin names.

Message	Valid characters are alphanumeric characters with dash and underscore only.
To fix	You need to specify valid characters only.

qcrv32_rule_jtag_empty_pins (error)

Message	Empty pin names found: <Pin description names>
To fix	You need to specify the pin names.

qcrv32_rule_jtag_invalid_pins (error)

Message	Invalid pin names found: <pin description names>
To fix	Valid characters are alphanumeric characters with dash and underscore only.

qcrv32_rule_mem_clk_resource (error)

Message	Memory Clock source is not configured
To fix	Assign a Memory Clock Source for the QCRV32 instance.

Message	PLL(<PLL Resource Name>) driving QCRV32's memory clock is not configured
To fix	You get this message if you did not create a PLL instance to drive the RISC-V memory clock source. Instantiate the correct PLL and clockout according to the memory clock source you use.

Message	PLL(<PLL Resource Name>).CLKOUT<*number*> driving QCRV32's memory clk is not configured
To fix	You get this message if you did not use the correct clockout of the PLL instance driving the RISC-V memory clock source. Instantiate the correct clockout according to the memory clock source you use.

qcrv32_rule_mem_clk_resource (warning)

Message	The memory clock frequency exceeds the maximum specification of {max_freq} MHz.
To fix	The maximum frequency depends on the device, timing model, and pipeline setting (**Quad-Core RISC-V block > Clock/Control tab > Enable the pipeline for SoC AXI memory interface**). Refer to the "DC and Switching Characteristics" topic in the data sheet for the maximum frequency.

qcrv32_rule_ocr_file (error)

Message	On-Chip-RAM file: <ocr_file_path> not exist
To fix	Add the file by clicking the **On-Chip RAM Configuration file** add file button in the **Base** tab of the Quad-Core RISC-V instance.

Message	Invalid On-Chip-RAM file format, only support intel hex/ bin
To fix	The On-Chip-RAM file format must be a **.hex** or a **.bin** file only.

Message	File too large, max size is {MAX_FILE_BYTES} bytes, got = {User file size}
To fix	The On-Chip-RAM file size must not be more than {MAX_FILE_BYTES}.

Message	Invalid Intel Hex record: <Detail message about error on line xxx>
To fix	The **.hex** file is corrupted. Choose a different file.

qcrv32_rule_resource (error)

Message	Resource is not a valid QCRV32 device instance.
To fix	Choose a resource in **SOC Resource** that exists in the device.

qcrv32_rule_sys_clk_resource (error)

Message	System Clock source is not configured
To fix	Assign a resource in **System Clock Source** > **Quad-Core RISC-V block** > **Clock/Control tab** that exists in the device.

Message	PLL(<PLL Resource Name>) driving QCRV32's system clock is not configured
To fix	You get this message if you did not create a PLL instance to drive the RISC-V system clock source. Instantiate the correct PLL and clockout according to the system clock source you use.

Message	PLL(<PLL Resource Name>).CLKOUT<*number*> driving QCRV32's system clk is not configured
To fix	You get this message if you did not use the correct clockout of the PLL instance driving the RISC-V system clock source. Instantiate the correct clockout according to the system clock source you use.

qcrv32_rule_sys_clk_resource (warning)

Message	The system clock frequency exceeds the maximum specification of {max_freq} MHz.
To fix	The maximum frequency depends on the device, timing model, and pipeline setting (**Quad-Core RISC-V block** > **Clock/Control tab** > **Enable the pipeline for SoC AXI memory interface**). Refer to the "DC and Switching Characteristics" topic in the data sheet for the maximum frequency.

qcrv32_rule_pll_non_frac (error)

Message	PLL {pll_inst_name} driving QCRV32 should disable fractional mode
To fix	Disable fractional mode for the PLL instance driving the QCRV32 instance's clock source.

Message	Found {number of pins} QCRV32 pin driven by PLL in fractional mode: {Pin names}
To fix	Disable fractional mode for the PLL instance driving those pins connected to QCRV32 instance's AXI, periphery, or JTAG clock input.

Chapter 14

SPI Flash Interface

Contents:

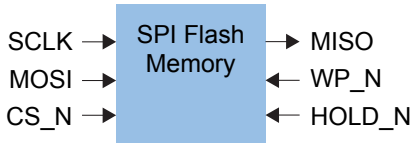
Titanium Ti35 and Ti60 FPGAs in the F100S3F2 package have an internal SPI flash memory.

About the SPI Flash Memory

Titanium FPGAs in the F100S3F2 package include a SPI flash memory. The SPI flash memory has a density of 16 Mbits and a clock rate of up to 85 MHz. In active configuration mode, the FPGA is configured using the configuration bitstream in the SPI flash memory. Typically you can fit two compressed bitstream images into the F100S3F2 SPI flash.

Important: You cannot enable the Titanium FPGA security features when using compressed bitstreams.

Figure 59: SPI Flash Memory Block Diagram



Important: The SPI flash memory's VCC is connected to `VCCIO1A_4B`. If you are using the SPI flash memory, drive the `VCCIO1A_4B` with a 1.8 V supply.

Table 122: SPI Flash Memory Signals (Interface to FPGA Fabric)

SPI Name	Signal	Direction	Description
SCLK	SCLK_OUT	Input	Clock output from FPGA CCK pin to SPI flash memory.
	SCLK_OE	Input	Output enable. Required for multiple controller.
MOSI	MOSI_IN	Output	Required for x2 or x4 data width.
	MOSI_OUT	Input	Data output from FPGA CDI0 to SPI flash memory.
	MOSI_OE	Input	Output enable. Required for x2 data width, x4 data width, or multiple controller.
MISO	MISO_IN	Output	Data input to FPGA CDI1 from SPI flash memory.
	MISO_OUT	Input	Required for x2 or x4 data width.
	MISO_OE	Input	Output enable. Required for x2 or x4 data width.
WP_N	WP_N_IN	Output	Required for x4 data width.
	WP_N_OUT	Input	Data output from FPGA CDI2 pin to SPI flash memory.
	WP_N_OE	Input	Output enable. Required for x4 data width or multiple controller.

SPI Name	Signal	Direction	Description
HOLD_N	HOLD_N_IN	Output	Required for x4 data width.
	HOLD_N_OUT	Input	Data output from FPGA CDI3 pin to SPI flash memory
	HOLD_N_OE	Input	Output enable. Required for x4 data width or multiple controller.
CS_N	CS_N_OUT	Input	Chip select output from FPGA SSL_N pin to SPI flash memory.
	CS_N_OE	Input	Output enable. Required for multiple controller.
CLK	CLK	Input	Required for register interface.

To program the Titanium F100S3F2 the SPI flash memory, use one of these modes:

- SPI Active using JTAG Bridge (New) mode
- SPI Active mode

Learn more: Refer to the AN 033: Configuring Titanium FPGAs for information on programming the SPI flash memory.

The GPIOL_P_01 (SSL_N), GPIOL_N_01 (CCK), GPIOL_P_03 (CDI0), and GPIOL_N_03 (CDI1) resources are for the SPI active interface. You can use these signals to read/write user data to/from the SPI flash memory while the Titanium FPGA is in user mode. You enable this feature by adding the SPI flash block to your interface design. These resources are not available as user I/O pins if you use the SPI flash block.

You can also write a new bitstream to the SPI flash memory by controlling the SPI signals with an external controller. In this case, the CRESET_N signal should stay low and the FPGA remains in reset mode, even though you stored a new bitstream in the SPI flash memory. To enable this mode, turn on **Configuration > External Flash Control > Enable external controller acccess to flash memory** in the Interface Designer.

Table 123: SPI Flash Interface Designer Settings

Option	Choices	Notes
Instance Name	User defined	
SPI Flash Resource	SPI_FLASH0	Only one resource available.
Enable Register Interface	0, 1	Default: 0 (Disable)
Read/Write Width	x1, x2, x4	Default: x1
Enable Multiple Controller	0, 1	Default: 0 (Disable)
Pin names (various)	User defined	Specify the interface pin names.

Using the SPI Flash Interface

The internal SPI flash memory is 16 Mbits and can hold:

- 1 uncompressed bitstream or
- 2 compressed bitstreams (typical designs) or
- 1 compressed bitstream and user data

Note: The maximum bitstream size for Ti35 and Ti60 FPGAs is about 13.7 Mbits; compression typically reduces the size by about 50%. So you have about half of the flash left over for user data if you only store one compressed bitstream.

If you want to use the internal SPI flash memory to store user data, you need to add the SPI flash interface block to your interface design. Simply add the block, choose the resource, and specify the instance and pin names. Then, connect the pins to your user design. Only use the SPI flash interface block to communicate with the internal SPI flash memory in user mode; you do not use this block for external flash devices.

Important: You **do not** need to use the SPI flash interface block if you are **only** using the internal SPI flash for storing bitstreams.

The following table lists the SPI flash interface block and the internal SPI flash memory signals with the default resource assignments.

Table 124: SPI Flash Resource Assignments

SPI Flash Interface Signal	SPI Flash Signal	F100S3F2 Package Resource Assignment
SCLK	SCK	GPIOL_N_01_CCK
MOSI	SI	GPIOL_P_03_CDI0
MISO	SO	GPIOL_N_03_CDI1
WP_N	WP#	_[15]
HOLD_N	HOLD#	_[15]
CS_N	CS#	GPIOL_P_01_SSL_N

[15] WP_N and HOLD_N signals are not bonded out.

Design Check: SPI Flash Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

spi_flash_rule_resource (error)

Message	Resource name is empty
To fix	Assign instance to a valid SPI Flash resource.

Message	Resource is not a valid SPI Flash device instance.
To fix	Assign instance to a resource that exists in the device.

spi_flash_rule_usage (error)

Message	Use a different resource for the following instance as it conflicts with SPI Flash <SPI Flash instance name> resource usage: <Other instance name that has conflict with the SPI Flash resource>
To fix	Assign different resources to the instances that have a resource conflict with the SPI Flash.

spi_flash_rule_empty_pins (error)

Message	Empty pin names found: <Pin description names>
To fix	Assign the pin names.

spi_flash_rule_invalid_pins (error)

Message	Invalid pin names found: <pin description names>
To fix	Valid characters are alphanumeric characters with dash and underscore only.

spi_flash_rule_instance_count (error)

Message	There can only be one SPI Flash instance.
To fix	You cannot have more resources than the number of available resources in the device.

spi_flash_rule_reg_clk_pin (error)

Message	Clock Pin Name is required when Register Interface is enabled
To fix	Assign a pin name for the clock pin if you want to use the register interface.

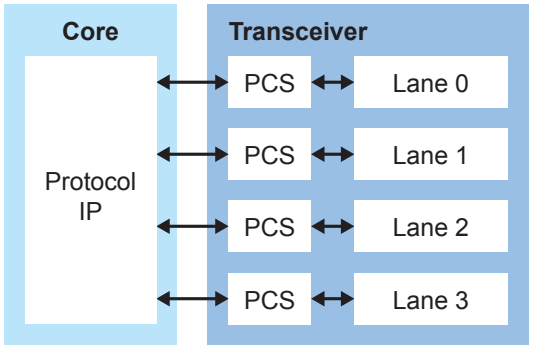
Chapter 15

Ethernet SGMII Interface

Contents:

The Ethernet SGMII high-speed transceiver interface supports 10 Mbps, 100 Mbps, and 1 Gbps MAC Ethernet data rates for 1 Gbps SGMII as well as 2.5 Gbps SGMII data rates. All four Titanium transceiver banks and lanes support SGMII.

Figure 60: Transceiver Used as Ethernet SGMII



The PCS supports Ethernet SGMII, 10GBase-KR, or PMA Direct for each lane

Each transceiver bank is called a "quad." Each quad has four lanes, and each lane supports an SGMII interface.

Ti375, Ti240, and Ti165 Titanium FPGAs have four quads available.

Ti85 and Ti135 Titanium FPGA have two quads available.

Note: The number of transceivers is package dependent. Refer to the data sheet for the number of transceivers available in the package you are using.

Ethernet SGMII Base Tab

In this tab you choose the resource and specify the instance name.

Table 125: Base Tab Settings

Where *n* is 0-3.

Parameter	Choices	Notes
Instance Name	User defined	Specify the instance name.
SGMII Resource	Q*n*_LN0 Q*n*_LN1 Q*n*_LN2 Q*n*_LN3	Choose the resource. You cannot use the same quad for SGMII and PCIe. The software prompts you with a message if you try to choose a quad that is already in use.
Quad-Lane Settings	Button	Click this button to open the **Quad-Lane Settings** dialog box that has options that apply to all lanes in the transceiver quad (e.g., reference clocks). Refer to Quad-Lane Settings on page 221.

Some settings apply to all lanes in a quad. The software prompts you with a message when you change the resource to a lane in another quad.

Ethernet SGMII Control Register Tab

In this tab you specify control settings.

Table 126: Control Register Tab Settings

Parameter	Choices	Notes
Enable Activity Status for LED	On, off	Default: off. Turn on to enable LED status activity.
Enable SGMII Auto Negotiation (AN)	On, off	Default: on. Turn on to enable auto negotiation.
Speed	10/100/1000 Mbps 2.5 Gbps	Default: 10/100/1000 Mbps.

Ethernet SGMII Pins Tab

This tab has several sub-tabs for defining the Ethernet SGMII interface pins.

Table 127: Pins Tab: Clock and Reset Sub-Tab Settings

Parameter	Choices	Notes
Interface Clock Input Connection Type	gclk, rclk	Default: rclk. Choose whether to use a global clock (gclk) or regional clock (rclk).
Interface Clock X2 Input Connection Type	rclk, gclk	Default: rclk. Choose whether to use a global clock (gclk) or regional clock (rclk) for the X2 clock.
Interface Clock Input Pin Name	User defined	Specify the pin names. Efinix recommends using the default names.
Interface Clock X2 Input Pin Name		
PCS Receive Reset Pin Name		
PCS Transmit Reset Pin Name		
PHY Lane Reset Pin Name		

Table 128: Pins Tab: Control Sub-Tab

Parameter	Selection	Notes
PMA Transmit Electrical Idle Pin Name	User defined	Efinix recommends using the default names.

Table 129: Pins Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
CTC Error Pin Name	User defined	Define the pin names for the error and status signals. Efinix recommends using the default names.
PHY Interrupt Pin Name		
Sync Status Pin Name		

Table 130: Pins Tab: Power Up Sub-Tab Settings

Parameter	Selection	Notes
Link PLL Clock Enable Acknowledge Pin Name	User defined	Define the pin names for the signals used when the interface powers up. Efinix recommends using the default names.
Link PLL Clock Enable Pin Name		
Link Power State Acknowledge [3:0] Bus Name		
Link Power State Request [3:0] Bus Name		
PMA Receiver Signal Detect Pin Name		

Table 131: Pins Tab: SGMII Sub-Tab Settings

Parameter	Selection	Notes
PCS Auto Negotiation Complete Pin Name	User defined	Define the pin names for the signals used for the SGMII signals. Efinix recommends using the default names.
Receive GMII Control [1:0] Bus Name		
Receive GMII Data [15:0] Bus Name		
Receive GMII Error [1:0] Bus Name		
SGMII Mode [1:0] Bus Name		
Transmit GMII Data [15:0] Bus Name		
Transmit GMII Enable [1:0] Bus Name		
Transmit GMII Error [1:0} Bus Name		

Ethernet SGMII Common Properties Tab

The settings in this tab apply to all lanes in the same quad.

Table 132: Common Properties Tab: Config Sub-Tab Settings

Parameter	Choices	Notes
Common Instance Name	User defined	Default: cmn_inst1. This field defines the prefix that is applied to all common signals.
Remove 1 Preamble Byte for Even Number of Idles	On, off	Default: off.

Table 133: Common Properties Tab: APB Sub-Tab

Parameter	Selection	Notes
APB Clock Pin Name	User defined	Specify the pin name.
Invert APB Clock Pin	On, off	Default: off. Indicate whether to invert the clock signal.
Pin and bus names	User defined	Define the APB pin names. Efinix recommends using the default names.

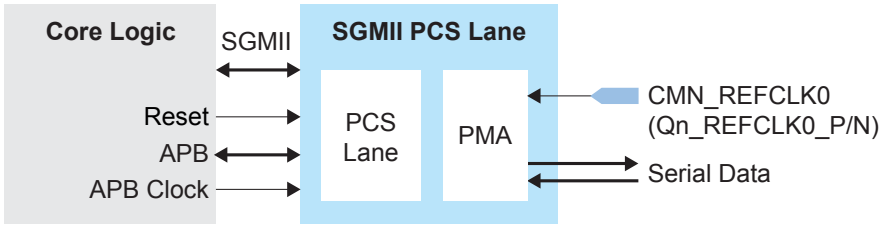
Table 134: Common Properties Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
PHY Ready Pin Name	User defined	Define the pin name. Efinix recommends using the default name.

Using the Ethernet SGMII Interface

You use the Efinity Interface Designer to implement an Ethernet SGMII block and configure it. The `CMN_REFCLK0` pin is a dedicated package pin.

Figure 61: Interface Designer Block Diagram



Tip: The Ethernet SGMII interface is complicated. If you find yourself turning a lot of options on and off to enable or disable features, some of the GUI menus might get out of sync. Therefore, Efinix suggests that you note the settings you want, and then start fresh with a newly created SGMII block with only the settings you want.

Design Check: Ethernet SGMII Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

sgmii_rule_external_clock (error)

Message	PLL LC0 is fixed to be driven by reference clock 0
To fix	In the <*project*>.**peri.xml** file, update the value of `PMA_CMN__cmn_plllc_gen_preg__cmn_plllc_pfdclk1_sel_preg` to `Refclk 0`.

sgmii_rule_hw_drc (error)

Message	Invalid value assigned to the following parameters: <list_of_parameters> Found <#> HW Errors: <list_of_errors>
To fix	Enter valid values for the parameters.

sgmii_rule_hw_drc (warning)

Message	Found <#> HW Warnings: <list_of_errors>
To fix	Enter valid values for the parameters.

sgmii_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

sgmii_rule_interface_clock (error)

Message	Interface Clock Input pin name cannot be empty
To fix	Add a pin name for the instance in the **Block Editor** > **Pins** > **Interface Clock Input Pin Name** box.

sgmii_rule_invalid_hex_value (error)

Message	The following hexadecimal parameters has invalid value: *<list of parameters with error message>*
To fix	Update the parameters to use a hexadecimal value in a valid range.

sgmii_rule_osc_clock (error)

Message	Oscillator is required to be configured
To fix	Create an Oscillator instance with no enable pin.

sgmii_rule_resource (error)

Message	Resource name is empty Resource is not a valid Ethernet SGMII device instance
To fix	Assign a valid resource.

sgmii_rule_res_usage (error)

Message	Resource name conflict with instance <instance name>
To fix	The resource is already used. Choose another one.

Message	Resource conflicts with PCI Express resource <resource name>
To fix	You cannot use the same resource for PCI Express and SGMII at the same time. Update one of the resources. (PCI Express uses quad 0 and quad 2.)

common_quad_lane_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

common_quad_lane_rule_pin_name (error)

Message	The following pin(s) is/are identical across different QUAD: *<list of pin and resource names>*
To fix	Check the pin names to in the **Block Editor** > **Common Properties tab** to ensure that you are not using the same name(s) as the lane-specific pin(s).

common_quad_rule_apb_clock (error)

Message	APB Clock pin cannot be empty APB Clock pin cannot be empty when APB is enabled
To fix	Enter the clock name in the **Block Editor** > **Common Properties** > **APB tab** or turn off the **Enable Advanced Peripheral Bus** option.

common_quad_rule_core_refclk_pin (error)

Message	Empty core reference clock pin name for: <*pin name*>
To fix	Specify the core clock pin name as the reference clock in the **Quad-Lane Settings** dialog box.

common_quad_rule_on_board_crystal (error)

Message	Reference clock should be from on-board crystal when using <resource name>
To fix	Assign one of the lane-based instances to Q0 or Q2.

common_quad_rule_pll_common_setting (error)

Message	Mismatch settings on the common PLL parameters across lanes: <*list of clocks and lanes*>
To fix	Transceivers in the same quad need to use the same PLL settings.

common_quad_rule_refclk (error)

Message	Found # lane instances using disabled Refclk 1
To fix	Open the <*project name*>.**peri.xml** and change the value of `REFCLK_SEL` to `Refclk 0`.

Message	Refclk 0 frequency invalid for instances: <list of instances> and Refclk 1 frequency invalid for instances: <list of instances> Refclk 0 frequency invalid for instances: <list of instances> Refclk 1 frequency invalid for instances: <list of instances>
To fix	Update reference clock frequency or use a different reference clock for the instances that are shown as invalid in the **Quad-Lane Settings** dialog box **Console**.

Message	PLL instance with resource <PLL resource list> is required to be configured when reference clock is not from on-board crystal
To fix	Create a PLL instance and assign one of the resource listed in the message, or turn on the **Reference clock from on-board crystal** option in the **Quad-Lane Settings** dialog box.

Message	Either one of the following PLL instance(s) required to enable output clock when reference clock is not from on-board crystal: <list of PLL instances with expected output clock>
To fix	Enable the output clock for one of the PLL instances.

Message	Either one of the following PLL instance(s) required to use <list of reference clock source> as exteranl reference clock source when PCIe reference clock is not from on-board crystal: <list of PLL instances>
To fix	If the reference clock is **not** from the on-board crystal, you need to set up an external reference clock. Add a PLL block and assign it to one of the listed PLL resources. Then, add a GPIO block as the PLL's reference clock and assign it to the resource shown in the PLL block.

Message	Either one of the following PLL instance(s) required to use local feedback mode when reference clock is not from on-board crystal: <list of PLL instances>
To fix	Change feedback mode to local for either one of the listed PLL instances.

common_quad_rule_refclk_usage (error)

Message	At least 1 lane-based instance have to use reference clock 0
To fix	For the transceivers in the same quad, at least one of the lanes must use reference clock 0. Use the **Quad-Lane Settings** dialog box to adjust the settings.

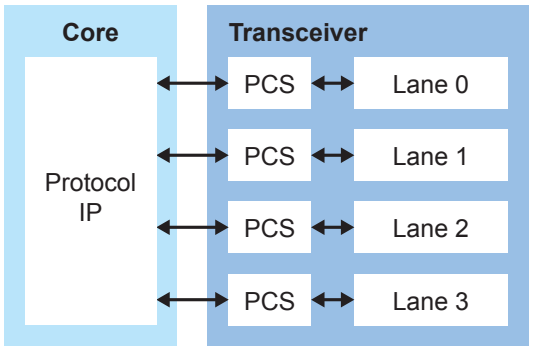
Chapter 16

Ethernet XGMII Interface

Contents:

The Ethernet XGMII high-speed transceiver interface supports 10.3125 Gbps data rates. All four Titanium transceiver banks and lanes support 10GBase-KR per IEEE Std. 802.3ap-2007.

Figure 62: Transceiver Used as Ethernet XGMII



The PCS supports Ethernet SGMII, 10GBase-KR, or PMA Direct for each lane

Each transceiver bank is called a "quad." Each quad has four lanes, and each lane supports an Ethernet XGMII interface.

Ti375, Ti240, and Ti165 Titanium FPGAs have four quads available.

Ti85 and Ti135 Titanium FPGA have two quads available.

Note: The number of transceivers is package dependent. Refer to the data sheet for the number of transceivers available in the package you are using.

Ethernet XGMII Base Tab

In this tab you choose the resource and specify the instance name.

Table 135: Base Tab Settings

Where *n* is 0-3.

Parameter	Choices	Notes
Instance Name	User defined	Specify the instance name.
XGMII Resource	Q*n*_LN0 Q*n*_LN1 Q*n*_LN2 Q*n*_LN3	Choose the resource. You cannot use the same quad for Ethernet XGMII and PCIe. The software prompts you with a message if you try to choose a quad that is already in use.
Quad-Lane Settings	Button	Click this button to open the **Quad-Lane Settings** dialog box that has options that apply to all lanes in the transceiver quad (e.g., reference clocks). Refer to Quad-Lane Settings on page 221.

Some settings apply to all lanes in a quad. The software prompts you with a message when you change the resource to a lane in another quad.

Ethernet XGMII Control Register Tab

In this tab you specify control settings.

Table 136: Control Register Tab Settings

Parameter	Choices	Notes
Enable Forward Error Correction (FEC)	On, off	Default: off. Turn on to enable FEC.
Enable Auto Negotiation (AN) Clause 37	On, off	Default: on. Turns on auto negotiation. Only supported for 10 Gbps speed.
Invert RX Polarity	On, off	Default: off. Turn on to invert the polarity of the RX.
Invert TX Polarity	On, off	Default: off. Turn on to invert the polarity of the TX.
Speed	5 Gbps, 10 Gbps	Default: 10 Gbps.
USXGMII AN Ordered set code	0x0 - 0xff	Default: 0x3. Define the ordered set that has the auto-negotiation information. Only supported for 10 Gbps speed.

Ethernet XGMII Pins Tab

This tab has several sub-tabs for defining the Ethernet XGMII interface pins.

Table 137: Pins Tab: Clock and Reset Sub-Tab Settings

Parameter	Choices	Notes
Interface Clock Input Connection Type	gclk, rclk	Default: rclk. Choose whether to use a global clock (gclk) or regional clock (rclk).
Interface Clock Pin Name	User defined	Specify the clock name.
PCS Receive Reset Pin Name	User defined	Specify the pin names. Efinix recommends using the default names.
PCS Transmit Reset Pin Name		
PHY Lane Reset Pin Name		

Table 138: Pins Tab: Control Sub-Tab

Parameter	Selection	Notes
Ethernet EEE Alert Enable Pin Name	User defined	Efinix recommends using the default names.
PMA Transmit Electrical Idle Pin Name	User defined	Efinix recommends using the default names.
Enable KR Base	On, off	Default: on. Turns on the KR base function.
KR Training tab	User defined	This tab defines the pin names for KR training. These pins are used when the **Enable KR Base** option is turned on. Efinix recommends using the default names.

Table 139: Pins Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the error and status signals. Efinix recommends using the default names.

Table 140: Pins Tab: Power Up Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the signals used when the interface powers up. Efinix recommends using the default names.

Table 141: Pins Tab: XGMII Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the signals used for the XGMII signals. Efinix recommends using the default names.

Ethernet XGMII Common Properties Tab

The settings in this tab apply to all lanes in the same quad.

Table 142: Common Properties Tab: Config Sub-Tab Settings

Parameter	Choices	Notes
Common Instance Name	User defined	Default: cmn_inst1. This field defines the prefix that is applied to all common signals.
Remove 1 Preamble Byte for Even Number of Idles	On, off	Turn on to remove a preamble byte.

Table 143: Common Properties Tab: APB Sub-Tab

Parameter	Selection	Notes
APB Clock Pin Name	User defined	Specify the pin name.
Invert APB Clock Pin	On, off	Default: off. Indicate whether to invert the clock signal.
Pin and bus names	User defined	Define the APB pin names. Efinix recommends using the default names.

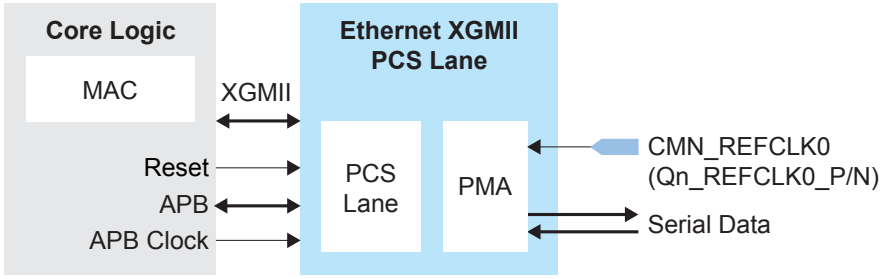
Table 144: Common Properties Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
PHY Ready Pin Name	User defined	Define the pin name. Efinix recommends using the default name.

Using the Ethernet XGMII Interface

You use the Efinity Interface Designer to implement an Ethernet XGMII block and configure it. The `CMN_REFCLK0` pin is a dedicated package pin.

Figure 63: Interface Designer Block Diagram



Tip: The Ethernet XGMII interface is complicated. If you find yourself turning a lot of options on and off to enable or disable features, some of the GUI menus might get out of sync. Therefore, Efinix suggests that you note the settings you want, and then start fresh with a newly created Ethernet XGMII block with only the settings you want.

Design Check: Ethernet XGMII Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

10gbase_kr_rule_external_clock (error)

Message	PLL LC0 is fixed to be driven by reference clock 0
To fix	In the <*project*>.**peri.xml** file, update the value of `PMA_CMN__cmn_plllc_gen_preg__cmn_plllc_pfdclk1_sel_preg` to `Refclk 0`.

10gbase_kr_rule_hw_drc (error)

Message	Invalid value assigned to the following parameters: <list_of_parameters> Found <#> HW Errors: <list_of_errors>
To fix	Enter valid values for the parameters.

10gbase_kr_rule_hw_drc (warning)

Message	Found <#> HW Warnings: <list_of_errors>
To fix	Enter valid values for the parameters.

10gbase_kr_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

10gbase_kr_rule_interface_clock (error)

Message	Interface Clock Input pin name cannot be empty
To fix	Add a pin name for the instance in the **Block Editor** > **Pins** > **Interface Clock Input Pin Name** box.

10gbase_kr_rule_invalid_hex_value (error)

Message	The following hexadecimal parameters has invalid value: <list_of_parameters_with_error_message>
To fix	Update the parameters to use a hexadecimal value in a valid range.

10gbase_kr_rule_osc_clock (error)

Message	Oscillator is required to be configured
To fix	Create Oscillator instance with no enable pin.

10gbase_kr_rule_resource (error)

Message	Resource name is empty
To fix	Assign a resource.

Message	Resource is not a valid Ethernet XGMII device instance
To fix	Assign the instance to a resource that exists in the device.

10gbase_kr_rule_res_usage (error)

Message	Resource name conflict with instance <instance name>
To fix	The resource is already used. Choose another one.

Message	Resource conflicts with PCI Express resource <resource name>
To fix	You cannot use the same resource for PCI Express and Ethernet XGMII at the same time. Update one of the resources. (PCI Express uses quad 0 and quad 2.)

common_quad_lane_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

common_quad_lane_rule_pin_name (error)

Message	The following pin(s) is/are identical across different QUAD: *<list of pin and resource names>*
To fix	Check the pin names to in the **Block Editor** > **Common Properties tab** to ensure that you are not using the same name(s) as the lane-specific pin(s).

common_quad_rule_apb_clock (error)

Message	APB Clock pin cannot be empty APB Clock pin cannot be empty when APB is enabled
To fix	Enter the clock name in the **Block Editor** > **Common Properties** > **APB tab** or turn off the **Enable Advanced Peripheral Bus** option.

common_quad_rule_core_refclk_pin (error)

Message	Empty core reference clock pin name for: *<pin name>*
To fix	Specify the core clock pin name as the reference clock in the **Quad-Lane Settings** dialog box.

common_quad_rule_on_board_crystal (error)

Message	Reference clock should be from on-board crystal when using <resource name>
To fix	Assign one of the lane-based instances to Q0 or Q2.

common_quad_rule_pll_common_setting (error)

Message	Mismatch settings on the common PLL parameters across lanes: *<list of clocks and lanes>*
To fix	Transceivers in the same quad need to use the same PLL settings.

common_quad_rule_refclk (error)

Message	Found # lane instances using disabled Refclk 1
To fix	Open the <*project name*>.**peri.xml** and change the value of `REFCLK_SEL` to `Refclk 0`.

Message	Refclk 0 frequency invalid for instances: <list of instances> and Refclk 1 frequency invalid for instances: <list of instances> Refclk 0 frequency invalid for instances: <list of instances> Refclk 1 frequency invalid for instances: <list of instances>
To fix	Update reference clock frequency or use a different reference clock for the instances that are shown as invalid in the **Quad-Lane Settings** dialog box **Console**.

Message	PLL instance with resource <PLL resource list> is required to be configured when reference clock is not from on-board crystal
To fix	Create a PLL instance and assign one of the resource listed in the message, or turn on the **Reference clock from on-board crystal** option in the **Quad-Lane Settings** dialog box.

Message	Either one of the following PLL instance(s) required to enable output clock when reference clock is not from on-board crystal: <list of PLL instances with expected output clock>
To fix	Enable the output clock for one of the PLL instances.

Message	Either one of the following PLL instance(s) required to use <list of reference clock source> as exteranl reference clock source when PCIe reference clock is not from on-board crystal: <list of PLL instances>
To fix	If the reference clock is **not** from the on-board crystal, you need to set up an external reference clock. Add a PLL block and assign it to one of the listed PLL resources. Then, add a GPIO block as the PLL's reference clock and assign it to the resource shown in the PLL block.

Message	Either one of the following PLL instance(s) required to use local feedback mode when reference clock is not from on-board crystal: <list of PLL instances>
To fix	Change feedback mode to local for either one of the listed PLL instances.

common_quad_rule_refclk_usage (error)

Message	At least 1 lane-based instance have to use reference clock 0
To fix	For the transceivers in the same quad, at least one of the lanes must use reference clock 0. Use the **Quad-Lane Settings** dialog box to adjust the settings.

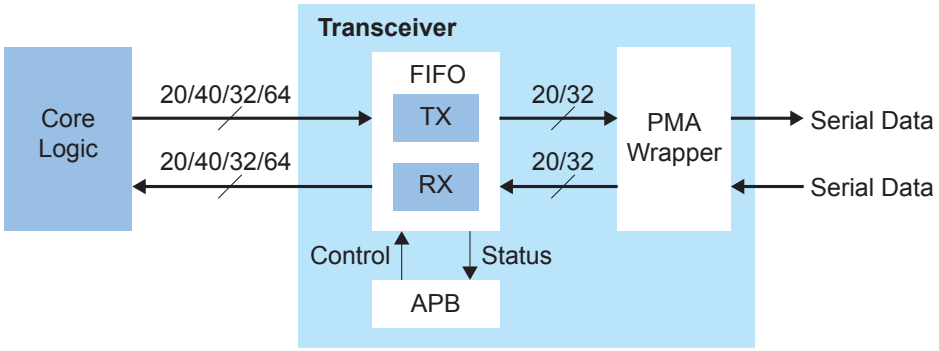
Chapter 17

PMA Direct Interface

Contents:

The PMA Direct mode uses the Titanium transceiver lane's PMA only. It supports serial data rates up to 12.5 Gbps. In this configuration, the serializer and deserializer interface connect directly to the FPGA fabric. The transceiver supports 20-, 40-, 32-, and 64-bit configurations. This mode gives you the flexibility to implement other transceiver protocols in the FPGA fabric.

Figure 64: Funtional Block Diagram



Note: The number of transceivers is package dependent. Refer to the data sheet for the number of transceivers available in the package you are using.

Tip: The PMA Direct interface is complicated. If you find yourself turning a lot of options on and off to enable or disable features, some of the GUI menus might get out of sync. Therefore, Efinix suggests that you note the settings you want, and then start fresh with a newly created PMA Direct block with only the settings you want.

PMA Direct Base Tab

In this tab you choose the resource and specify the instance name.

Table 145: Base Tab Settings

Where *n* is 0-3.

Parameter	Choices	Notes
Instance Name	User defined	Specify the instance name.
PMA Direct Resource	Q*n*_LN0 Q*n*_LN1 Q*n*_LN2 Q*n*_LN3	Choose the resource. You cannot use the same quad for PMA Direct and PCIe®. The software prompts you with a message if you try to choose a quad that is already in use.
Quad-Lane Settings	Button	Click this button to open the **Quad-Lane Settings** dialog box that has options that apply to all lanes in the transceiver quad (e.g., reference clocks). You also use this dialog box to choose presets or specify a custom combination of reference clock frequency, data rate, and SerDes bit width. See **Transceiver PLLs** on page 220 and **Quad-Lane Settings** on page 221.

The software prompts you with a message when you change the resource to a lane in another quad.

PMA Direct Control Register Tab

In this tab you specify control settings.

Table 146: Control Register Tab Settings

Parameter	Choices	Notes
Mode	"TX FIFO, RX FIFO" "TX FIFO, RX Register" TX FIFO RX FIFO	Choose whether you want a FIFO on RX and TX, or TX only (RX in register mode). Refer to "RX Register Mode (Design Considerations)" in the **Titanium PMA Direct User Guide** for details.
Bonding Mode	x1, x2, x4, x8	Choose how many lanes to use together. Refer to "Channel Bonding Mode" in the **Titanium PMA Direct User Guide** for details.
Tx equalization mode	Deemphasis, 3 taps FIR filter	Default: Deemphasis
Deemphasis	Off, 3.5 dB, 6 dB	Default: Off
Main C_0	User defined	Default: 0x2d Specify the coefficient values to use for the FIR filter. Used with **3 taps FIR filter** option.
Pre C_{-1}	User defined	Default: 0x0 Specify the coefficient values to use for the FIR filter. Used with **3 taps FIR filter** option.
Post C_{+1}	User defined	Default: 0x0 Specify the coefficient values to use for the FIR filter. Used with **3 taps FIR filter** option.

PMA Direct Pins Tab

This tab has several sub-tabs for defining the PMA Direct interface pins.

Table 147: Pins Tab: Clock and Reset Sub-Tab Settings

Parameter	Choices	Notes
Interface Transmit Clock Pin Name	User defined	Specify the clock name.
Transmit Clock Input Connection Type	gclk, rclk	Default: rclk. Choose whether to use a global clock (gclk) or regional clock (rclk).
Interface Receive Clock Pin Name	User defined	Specify the clock name.
Receive Clock Input Connection Type	gclk, rclk	Default: rclk. Choose whether to use a global clock (gclk) or regional clock (rclk).
PCS Receive Reset Pin Name	User defined	Specify the pin names. Efinix recommends using the default names.
PCS Transmit Reset Pin Name		
PHY Lane Reset Pin Name		

Table 148: Pins Tab: Control Sub-Tab

Parameter	Selection	Notes
PMA Transmit Electrical Idle Pin Name	User defined	Efinix recommends using the default name.

Table 149: Pins Tab: Data Interface Sub-Tab

Parameter	Selection	Notes
Receive Data [63:0] Bus Name	User defined	Efinix recommends using the default name.
Transmit Data [63:0] Bus Name	User defined	Efinix recommends using the default name.

Table 150: Pins Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
PHY Interrupt Pin Name	User defined	Define the pin name for the error and status signal. Efinix recommends using the default name.

Table 151: Pins Tab: Power Up Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the signals used when the interface powers up. Efinix recommends using the default names.

PMA Direct Common Properties Tab

The settings in this tab apply to all lanes in the same quad.

Table 152: Common Properties Tab: Config Sub-Tab Settings

Parameter	Choices	Notes
Common Instance Name	User defined	Default: cmn_inst1. This field defines the prefix that is applied to all common signals.

Table 153: Common Properties Tab: APB Sub-Tab

Parameter	Selection	Notes
Enable Advanced Peripheral Bus	On, off	Default: on. This setting allows you to use the APB to control the PMA Direct registers.
APB Clock Pin Name	User defined	Specify the pin name.
Invert APB Clock Pin	On, off	Default: off. Indicate whether to invert the clock signal.
Pin and bus names	User defined	Define the APB pin names. Efinix recommends using the default names.

Table 154: Common Properties Tab: Clock and Reset Sub-Tab Settings

Parameter	Selection	Notes
Enable PHY Quad Reset Pin	On, off	Default: off. Turn on to specify a reset pin name for the quad.
PHY Quad Reset Pin Name	User defined	Specify the reset pin name.

Table 155: Common Properties Tab: Error And Status Sub-Tab Settings

Parameter	Selection	Notes
PHY Ready Pin Name	User defined	Define the pin name. Efinix recommends using the default name.

PMA Direct Bonding Mode

When you use multiple PMA Direct lanes together, it is called bonding; the number of lanes bonded together is the bonding mode.

- *x1*—Only one lane is used by itself (with its own settings)
- *x2*—Use two lanes together (lanes 0 and 1 or lanes 2 and 3); they share settings.
- *x4*—Use all four lanes in the same quad together; they share settings.
- *x8*—Use eight lanes together; they share settings. In this case, you are using all lanes two quads.

You choose the mode with the **Bonding Mode** drop-down list box in the PMA Direct block's **Control Register** tab.

The software checks that you have set the bonding mode for enough lanes in the same quad to create the bond, e.g., two lanes are in x2 mode or 4 lanes are in x4 mode. If you choose x8, then you need to have all eight lanes in two quads set to x8.

Each PMA Direct block instance has reference clock, data rate, and SerDes width settings. When the lanes are bonded together, they need to use the same settings. To make it easier to copy the settings to all lanes that are bonded together, the software shows the **Apply Setting to <bonded lanes>** in the **Quad-Lane Settings** dialog box. When you click the button, the settings are copied to the other bonded lanes.

Note: Refer to Quad-Lane Settings on page 221 for more information on configuring common settings.

Design Check: PMA Direct Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

pma_direct_rule_bonding_mode (error)

Message	Bonding mode <x2,x4> requires <2,4> lanes configured in the same QUAD
To fix	Bonding mode x2 requires 2 lanes in the same quad to be configured; bonding mode x4 requires 4 lanes in the same quad to be configured

Message	Bonding mode x2 is valid on lane resource pairs LN0,LN1 or LN2,LN3 of the same QUAD only
To fix	Only certin lanes can be bonded together. Check that the lanes you are trying to bond are set to the correct resources. Refer to "Bonding Mode" in the **Titanium PMA Direct User Guide** for details.

Message	Mismatch properties against lane instances in the same bonding <bonding_mode>: <list of mismatched parameters>
To fix	Lanes that are bonded must have the same preperties. Update the listed parameters to ensure that the highlighted mismatched properties are identical for all PMA Direct lane instances within the same bond

Message	Bonding mode x8 requires all lanes in the quad to be configured
To fix	Create new instances and assign remaining resource in the same quad, or change the lane bonding mode. Bonding mode x8 requires you to configure all 4 lanes in 2 quads.

Message	Expected Q# lanes to be configured as part of bonding mode x8
To fix	Create new instances and assign resources for the quad, or change the bonding mode. Bonding mode x8 can only use quad pairs from the following list: (Q0, Q1), (Q2, Q3), (Q1, Q2).

Message	Bonding mode x8 requires 8 lane instances to be configured Bonding mode x8 requires 8 lane instances but found <number of instances>
To fix	Change the bonding mode for the other lane instance to ensure that there are 8 PMA Direct instances with bonding mode x8 for the correct quad-lane resource assignments For x8 mode, (Q0, Q1), (Q2, Q3), (Q1, Q2) are the available quad pairs.

Message	Clock resource is not selected for bonding mode <bonding mode>
To fix	Turn on the **Used as Clock Resource** option for one of the PMA Direct instances in the **Pins tab** > **Clock and Reset subtab**.

Message	Only 1 clock resource is allowed to be selected for bonding mode <bonding mode>, selected instances: <list of instances>
To fix	You can only use one PMA Direct clock as a clock signal to the core.Turn off the **Used as Clock Resource** for all the listed instances except for the one that you want to use as a clock resource.

pma_direct_rule_data_rate_timing_model (error)

Message	Data rate greater than <*max limit*> Gbps is not supported in timing model <*speed grade*>
To fix	Choose a lower data rate or select a device/timing model that supports the data rate. Refer to the "Transceiver Specifications" topic in the data sheet for supported rates by speed grade.

pma_direct_rule_external_clock (error)

Message	PLL LC0 is fixed to be driven by reference clock 0
To fix	In the <*project*>.**peri.xml** file, update the value of `PMA_CMN__cmn_plllc_gen_preg__cmn_plllc_pfdclk1_sel_preg` to `Refclk 0`.

pma_direct_rule_hw_drc (error)

Message	Invalid value assigned to the following parameters: <list_of_parameters> Found <#> HW Errors: <list_of_errors>
To fix	Enter valid values for the parameters.

pma_direct_rule_hw_drc (warning)

Message	Found <#> HW Warnings: <list_of_errors>
To fix	Enter valid values for the parameters.

pma_direct_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

pma_direct_rule_invalid_hex_value (error)

Message	The following hexadecimal parameters has invalid value: <list_of_parameters_with_error_message>
To fix	Update the parameters to use a hexadecimal value in a valid range.

pma_direct_rule_osc_clock (error)

Message	Oscillator is required to be configured
To fix	Create an Oscillator instance with no enable pin.

pma_direct_rule_preset (error)

Message	Preset reference clock frequency, <*frequency value*> does not match with common refclk frequency <*frequency value*>
To fix	Update the preset reference clock frequency or the common reference clock frequeny in the **Quad-Lane Settings** dialog box.

Message	Preset has invalid PLL lane config settings. Please reassign the preset: <*list of lane params*>
To fix	Use a different preset in the **Quad-Lane Settings** dialog box.

Message	Invalid PMA Direct preset with combination of Data Rate: <*data rate*>, SerDes Width: <*serdes width*> and Reference Clock Frequency: <*reference clock freq*> MHz
To fix	Change the reference clock frequeny or choose a different preset in the **Quad-Lane Settings** dialog box.

Message	Internal Eror: No PMA Direct PLL configuration map
To fix	Contact your local Efinix representative for help.

pma_direct_rule_rclk (error)

Message	Use global clock due to unroutable clock loopback on regional clock with bonding mode:<bonding_mode> Mode: <mode> on pins: <list of pin types>
To fix	Change the connection type for the listed pin types

pma_direct_rule_resource (error)

Message	Resource name is empty
To fix	Assign a resource.

Message	Resource is not a valid PMA Direct device instance
To fix	Assign the instance to a resource that exists in the device.

pma_direct_rule_res_usage (error)

Message	Resource name conflict with instance <instance name>
To fix	The resource is already used. Choose another one.

Message	Resource conflicts with PCI Express resource <resource name>
To fix	You cannot use the same resource for PCI Express and PMA Direct at the same time. Update one of the resources. (PCI Express uses quad 0 and quad 2.)

pma_direct_rule_rx_clock (error)

Message	Interface Receive Clock Input pin name cannot be empty on lane with clock resource enabled
To fix	Add a pin name for the instance in the **Block Editor** > **Pins** > **Interface Receive Clock Input Pin Name** box.

pma_direct_rule_rx_register_bonding_mode (error)

Message	Rx Register mode only supports bonding mode x1
To fix	For PMA Direct blocks, if **Control Register tab** > **Mode** is set to **TX FIFO, RX Register**, you cannot use x2, x4, or x8 bonding modes. Instead, choose **x1** for the **Bonding Mode** option.

pma_direct_rule_tx_clock (error)

Message	Interface Transmit Clock Input pin name cannot be empty
To fix	Add a pin name for the instance in the **Block Editor** > **Pins** > **Interface Transmit Clock Input Pin Name** box.

Message	Interface Transmit Clock Input cannot be used with bonding mode x8
To fix	In x8 mode you cannot use the TX pins. Choose **Mode** > **RX FIFO**, which filters out the TX pins. Clear the Interface Transmit Clock Pin Name field.

pma_direct_rule_x8_mode (error)

Message	Bonding mode x8 is only supported in Mode RX FIFO
To fix	Change the bonding mode and the mode.

pma_direct_rule_x8_mode_rx_clk_conn_type (error)

Message	Bonding mode x8 does not support rclk clock input connection type
To fix	Change the RX clock input connection type and bonding mode.

pma_direct_rule_x8_quad_pair (error)

Message	Bonding mode x8 on quad Q# expected to pair up with instances of one of the following quads: <list of quad names> Bonding mode x8 on quad Q# expected to pair up with quad Q#
To fix	Change the bonding mode for the listed quads, change the bonding mode of the current instance, or use a different quad resource.

Message	Bonding mode x8 requires 2 quads with PMA Direct configuration
To fix	Change the bonding mode for the other quad resource, or create an instance of another quad resource that adds up to 8 lanes in the bond.

common_quad_lane_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

common_quad_lane_rule_pin_name (error)

Message	The following pin(s) is/are identical across different QUAD: *<list of pin and resource names>*
To fix	Check the pin names to in the **Block Editor** > **Common Properties tab** to ensure that you are not using the same name(s) as the lane-specific pin(s).

common_quad_rule_apb_clock (error)

Message	APB Clock pin cannot be empty APB Clock pin cannot be empty when APB is enabled
To fix	Enter the clock name in the **Block Editor** > **Common Properties** > **APB tab** or turn off the **Enable Advanced Peripheral Bus** option.

common_quad_rule_core_refclk_pin (error)

Message	Empty core reference clock pin name for: *<pin name>*
To fix	Specify the core clock pin name as the reference clock in the **Quad-Lane Settings** dialog box.

common_quad_rule_on_board_crystal (error)

Message	Reference clock should be from on-board crystal when using <resource name>
To fix	Assign one of the lane-based instances to Q0 or Q2.

common_quad_rule_pll_common_setting (error)

Message	Mismatch settings on the common PLL parameters across lanes: *<list of clocks and lanes>*
To fix	Transceivers in the same quad need to use the same PLL settings.

common_quad_rule_refclk (error)

Message	Found # lane instances using disabled Refclk 1
To fix	Open the <*project name*>.**peri.xml** and change the value of `REFCLK_SEL` to `Refclk 0`.

Message	Refclk 0 frequency invalid for instances: <list of instances> and Refclk 1 frequency invalid for instances: <list of instances> Refclk 0 frequency invalid for instances: <list of instances> Refclk 1 frequency invalid for instances: <list of instances>
To fix	Update reference clock frequency or use a different reference clock for the instances that are shown as invalid in the **Quad-Lane Settings** dialog box **Console**.

Message	PLL instance with resource <PLL resource list> is required to be configured when reference clock is not from on-board crystal
To fix	Create a PLL instance and assign one of the resource listed in the message, or turn on the **Reference clock from on-board crystal** option in the **Quad-Lane Settings** dialog box.

Message	Either one of the following PLL instance(s) required to enable output clock when reference clock is not from on-board crystal: <list of PLL instances with expected output clock>
To fix	Enable the output clock for one of the PLL instances.

Message	Either one of the following PLL instance(s) required to use <list of reference clock source> as exteranl reference clock source when PCIe reference clock is not from on-board crystal: <list of PLL instances>
To fix	If the reference clock is **not** from the on-board crystal, you need to set up an external reference clock. Add a PLL block and assign it to one of the listed PLL resources. Then, add a GPIO block as the PLL's reference clock and assign it to the resource shown in the PLL block.

Message	Either one of the following PLL instance(s) required to use local feedback mode when reference clock is not from on-board crystal: <list of PLL instances>
To fix	Change feedback mode to local for either one of the listed PLL instances.

common_quad_rule_refclk_usage (error)

Message	At least 1 lane-based instance have to use reference clock 0
To fix	For the transceivers in the same quad, at least one of the lanes must use reference clock 0. Use the **Quad-Lane Settings** dialog box to adjust the settings.

common_quad_rule_phy_reset_pin (error)

Message	PHY Quad Reset pin cannot be empty when PHY Quad Reset Pin is enabled
To fix	Enter the reset pin name in the **PMA Direct block** > **Common Properties tab** > **Clock and Reset sub-tab** > **PMA Quad Reset Pin Name** box or turn off the **Enable PHY Quad Reset Pin** option.

common_quad_rule_pma_direct_x8_refclk (error)

Message	Refclk usage in x8 bonding quads not identical: QUAD_# and Quad_# Refclk frequencies in x8 bonding quads not identical: QUAD_# and QUAD_#
To fix	For x8 mode, all lanes in the quads need to use the same reference clock frequency. Update the reference clock frequency for one of the quads.

Chapter 18

Transceiver Interfaces (Common Settings)

Contents:

Portions of the transceiver interfaces are common (or shared) among protocols, for example the internal transceiver PLLs. If you are using the transceiver for PCIe, you cannot use it for another protocol at the same time. With the Ethernet and PMA Direct protocols however, you can combine them in a quad, using some lanes for one protocol and other lanes for a second one. When sharing the transceiver among protocols, you need to be aware of common settings and rules as described in the following topics.

Transceiver PLLs

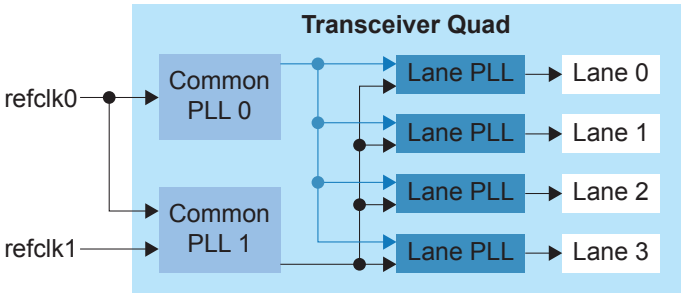
The transceiver quads have several PLLs:

- Two common PLLs that are shared by all lanes in the quad
- Four lane-based PLLs, one for each lane

These PLLs are driven by one or more reference clocks, `refclk0` and `refclk1`. `refclk0` can feed both common PLLs, while `refclk1` only feeds common PLL 1. You specify the reference clocks in the **Quad-Lane Settings** dialog box (see Quad-Lane Settings on page 221).

Note: The **Quad-Lane Settings** dialog box only applies for PMA Direct and Ethernet protocols. You do not use it for PCI Express blocks.

Figure 65: Transceiver Quad PLL Overview



The following figures show the block diagrams for the common and lane PLLs.

Figure 66: Common PLL Block Diagram

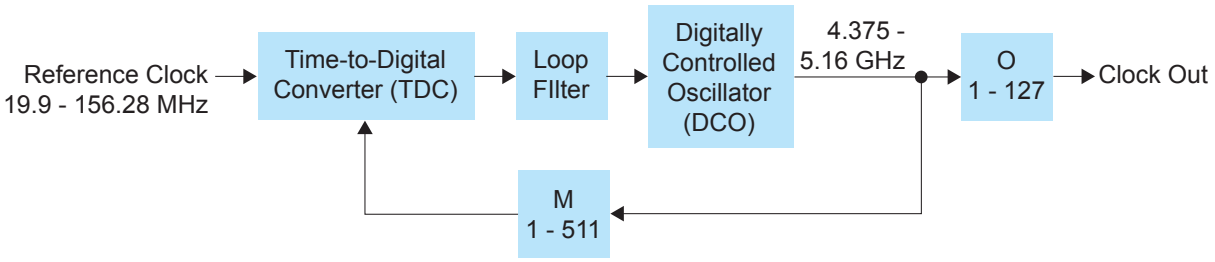
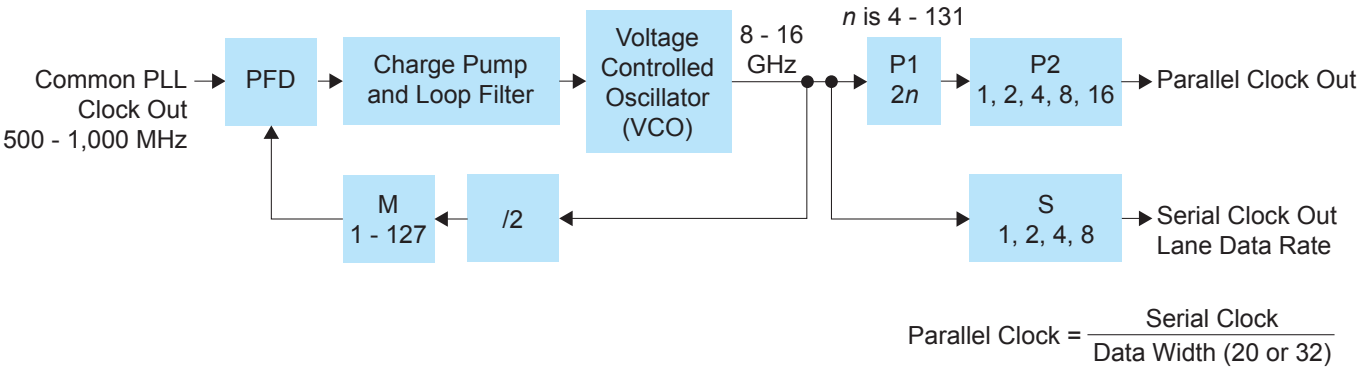


Figure 67: Lane PLL Block Diagram



Note: You cannot set the M, O, P1, P2, or S values. The software calculates them based on the reference clock, data rate, and SerDes width.

Because the transceiver protocols have different PLL requirements, and they share the common PLLs, there are some restrictions on which protocols you can use together in the same quad. All of the lanes in the same transceiver quad must have the same common PLL settings. For example:

- A PMA Direct lane with a 100-Mhz reference clock, 1.25-Gbps data rate, and 20-bit SerDes width *can* share the same common PLL as the Ethernet protocols because they all use the same common PLL settings (M = 50 and O = 6).
- A PMA Direct lane with a 100-Mhz reference clock, 2.7-Gbps data rate, and 20-bit SerDes width *cannot* share the same common PLL as the Ethernet protocols because the common PLL settings are different (M = 48 and O = 4 for PMA Direct and M = 50 and O = 6 for Ethernet).

Because the transceiver quad has two common PLLs, you can have protocols with mismatched common PLL settings in the same quad as long as you do not need more than two common PLLs total. For example, a PMA Direct lane with a 100-Mhz reference clock, 2.7-Gbps data rate, and 20-bit SerDes width *can* share the same *transceiver quad* as the Ethernet protocols as long as both common PLLs are available (one for PMA Direct and one for Ethernet).

Quad-Lane Settings

Each transceiver quad has four lanes that support Ethernet SGMII, Ethernet 10GBase-KR, and PMA Direct. You do not have to use the same protocol for all of the lanes in a quad: you can mix and match different protocols. For example, you can use PMA Direct for two lanes and Ethernet XGMII for two lanes. However, some settings must be the same for all lanes in

a quad. You make these settings in the **Quad Lane Settings** dialog box. Click the **Quad Lane Settings** button in the Ethernet SGMII, Ethernet XGMII, or PMA Direct block **Base** tab to open the dialog box.

Figure 68: Quad-Lane Settings Dialog Box

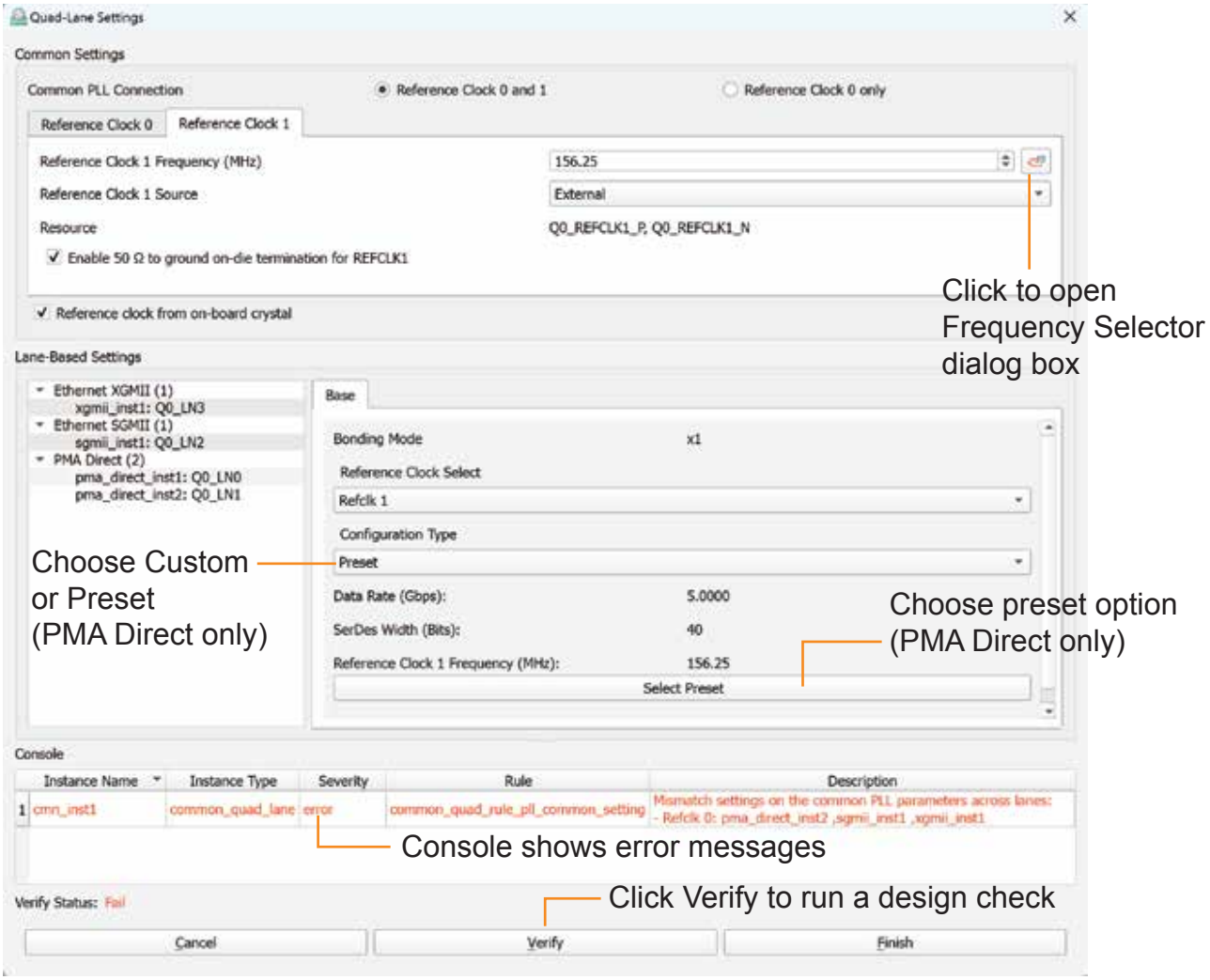


Table 156: Common PLL Connection Settings

Where *n* is 0 or 1.

Parameter	Choices	Notes
Reference Clock 0 and 1	On, off	Turn on to use two reference clocks (0 and 1) for the transceiver quad. In **Lane-Based Settings**, you need to specific which clock each transceiver block uses.
Reference Clock 0 only	On, off	Default. Use reference clock 0 as the clock for all lanes in the transceiver quad.
Reference Clock *n* Frequency (MHz)	Enter value or choose preset	Click the **Select reference clock frequency** button next to the box. The Frequency Selector dialog box opens with a list of valid frequencies for the different protocols. Choose one and click **OK**. You can also enter a value manually.
Reference Clock *n* Source	External	You must use an external clock. The Block Editor shows the name of the resource that you should use for this clock.

Parameter	Choices	Notes
Enable 50 Ω to ground on-die termination for REFCLK*n*	On, off	Default: on. Indicate whether to use on-die termination for the reference clock.

Table 157: Lane-Based Settings

Where *n* is 0, 1, 2, or 3.

Parameter	Choices	Notes
Reference Clock Select	Refclk0, Refclk 1	Choose the reference clock to use for the lane. **Refclk 1** is only available if you turn on the **Reference Clock 0 and 1** option.
Configuration Type	Preset, Custom	Preset: Choose from one of the presets for the data rate, SerDes width, and reference clock frequency. Custom: Specify the data rate, SerDes width, and reference clock frequency yourself.
Data Rate (Gbps)	1.25 to 12.5	Specify the data rate. Only available for the **Custom** configuration type. Default 12.5
SerDes Width (Bits)	20 bits, 32 bits, 40 bits, 64 bits	Choose the width. Only available for the **Custom** configuration type. Default: 40 bits
Calculate	–	After specifying the data rate and SerDes width, click **Calculate** to show the actual data rate that the transceiver can achieve. Only available for the **Custom** configuration type.

Parameter	Choices	Notes
Select Preset	1.25 Gbps-100.0 MHz-20 Bits 1.25 Gbps-100.0 MHz-40 Bits 1.485 Gbps-148.5 MHz-20 Bits 1.485 Gbps-148.5 MHz-40 Bits 2.376 Gbps-148.5 MHz-20 Bits 2.376 Gbps-148.5 MHz-40 Bits 2.5 Gbps-100.0 MHz-20 Bits 2.5 Gbps-100.0 MHz-40 Bits 2.7 Gbps-100.0 MHz-20 Bits 2.7 Gbps-100.0 MHz-40 Bits 2.97 Gbps-148.5 MHz-20 Bits 2.97 Gbps-148.5 MHz-40 Bits 3.125 Gbps-156.25 MHz-20 Bits 3.125 Gbps-156.25 MHz-40 Bits 4.752 Gbps-148.5 MHz-20 Bits 4.752 Gbps-148.5 MHz-40 Bits 5.0 Gbps-100.0 MHz-20 Bits 5.0 Gbps-100.0 MHz-40 Bits 5.0 Gbps-156.25 MHz-20 Bits 5.0 Gbps-156.25 MHz-40 Bits 5.4 Gbps-100.0 MHz-20 Bits 5.4 Gbps-100.0 MHz-40 Bits 5.94 Gbps-148.5 MHz-20 Bits 5.94 Gbps-148.5 MHz-40 Bits 6.25 Gbps-100.0 MHz-20 Bits 6.25 Gbps-100.0 MHz-40 Bits 6.375 Gbps-100.0 MHz-40 Bits 6.75 Gbps-100.0 MHz-40 Bits 8.0 Gbps-100.0 MHz-40 Bits 8.1 Gbps-50.0 MHz-40 Bits 8.1 Gbps-75.0 MHz-40 Bits 8.1 Gbps-90.0 MHz-40 Bits (continued)	PMA Direct only. Choose the data rate (in Gbps), reference clock frequency (in MHz), and SerDes width combination. Your selection displays in the GUI. Data rates higher than 10.3125 Gbps are only supported in C4, I4, C4L, I4L speed grades.
Select Preset (continued)	9.504 Gbps-148.5 MHz-40 Bits 10.0 Gbps-100.0 MHz-40 Bits 10.0 Gbps-156.25 MHz-40 Bits 10.3125 Gbps-156.25 MHz-40 Bits 11.88 Gbps-148.5 MHz-40 Bits 12.5 Gbps-100.0 MHz-40 Bits 12.5 Gbps-156.25 MHz-40 Bits	

Parameter	Choices	Notes
Apply Setting to *bonded lanes*	Click button	Use this button to copy all settings to the other PMA Direct transceiver lanes. This button only works for lanes that have the same bonding mode, and the bonding mode must be x2, x4, or x8. If the bonding modes are not the same, clicking the button has no effect. You set the bonding mode in the **Control Register** tab. Refer to PMA Direct Bonding Mode on page 213 for more details.

Click **Verify** to perform a design check. The **Console** shows any resulting error or warning messages.

Chapter 19

PCI Express® Interface

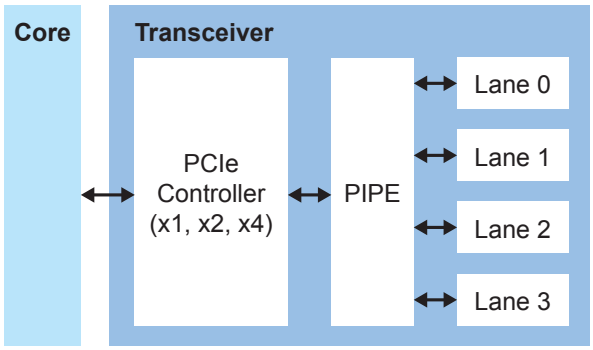
Contents:

The Titanium PCIe® interface supports data rates from 2.5 Gbps to 16 Gbps with x1, x2, or x4 configuration.

Table 158: Supported PCIe Standards

Standard	Data Rate (Gbps) per Lane	Number of Lanes	Specification
PCIe Gen 1	2.5	x1, x2, x4	PCI Express® Base Specification Revision 1.1
PCIe Gen 2	5	x1, x2, x4	PCI Express® Base Specification Revision 2.1
PCIe Gen 3	8	x1, x2, x4	PCI Express® Base Specification Revision 3.0
PCIe Gen 4	16	x1, x2, x4	PCI Express® Base Specification Revision 4.0

Figure 69: Transceiver Used for PCIe



Each transceiver bank is called a "quad." Titanium Ti375, Ti240, and Ti165 FPGAs have up to four quads available. Two transceiver quads support PCIe, QUAD_0 and QUAD_2.

Note: The number of transceivers is package dependent. Refer to the data sheet for the number of transceivers available in the package you are using.

PCI Express Base Tab

In the **Block Editor** > **Base** tab you choose the resource and specify high level settings.

Table 159: Base Tab Settings

Parameter	Choices	Notes
Instance Name	User defined	Specify the instance name
PCIe Resource	None, QUAD_0, QUAD_2	Choose the resource. QUAD_0 and QUAD_2 support PCIe.
Mode	Endpoint, Root Port	Choose whether the instance is an endpoint or root port. In v2024.1, the PCIe root port has limited functionality.
Link Width	x1, x2, x4	Choose the width.
Generation	Gen1 (2.5 Gbps) Gen2 (5.0 Gbps) Gen3 (8.0 Gbps) Gen4 (max 16.0 Gbps)	Choose the generation (and resulting speed).
Maximum Payload Size	128 bytes 256 bytes 512 bytes	Choose the payload size.
Gen3 Equalization RX Preset	User defined	Root port only. Default: 0x3
Gen3 Equalization TX Preset	User defined	Root port only. Default: 0x3
Gen4 Equalization TX Preset	User defined	Root port only. Default: 0x2
Reference Clock Frequency	User defined	Specify the frequency
Reference Clock Source	External	Defaults to external.
External Clock > Clock	CMN_REFCLK0	Defaults to CMN_REFCLK0. Block Editor shows the name of the resource that you should use for this clock.
Enable 50 Ω to ground on-die termination for REFCLK0	On, off	Default: on. Use termination for the reference clock.
Reference clock from on-board crystal	On, off	Default: on. Use a crystal on the board as the PCIe reference clock instead of a signal from the edge card connector. If you turn this option off, you need to create a PLL instance with: • BR0 or BR1 as the PLL resource • CLKOUT4 output clock enabled • An external reference clock resource (GPIOR_140 for BR0 or GPIOR_141 for BR1) • Local feedback mode This PLL provides the temporary PCIe reference clock while the PHY is configuring. When the PHY completes configuration, the reference clock reverts to the edge card connector.

PCI Express Reset Tab

This tab defines the reset signals.

The PCIe `PERST_N` signal triggers a warm reset. This signal comes from an I/O pad and requires you to instantiate a GPIO and assign it to the resource shown in the **Block Editor > Reset tab > PCI Express Reset box**. See GPIO Block (PERST_N) on page 234 for instructions on how to add the required GPIO block.

Table 160: Reset Tab Settings

Parameter	Choices	Notes
Hot Reset Input Pin Name	User defined	Only enabled in root port mode. Efinix recommends using the default name.
Hot Reset Output Pin Name	User defined	Efinix recommends using the default name.
Link Down Reset Pin Name	User defined	
Reset Acknowledge Pin Name	User defined	
Reset Request Pin Name	User defined	

PCI Express Function Tab

The **Function** tab is where you define the PCI ID for your application. The default vendor ID (**0x1f7a**) is Efinix's PCI SIG vendor ID number. In endpoint mode you can use up to 4 physical functions.

Table 161: Function Tab Settings (Endpoint Mode)

where *n* is 0-3.

Parameter	Choices	Notes
Subsystem Vendor ID	User defined	Default: 0x1f7a
Vendor ID	User defined	Default: 0x1f7a
Total Physical Functions	1-4	Default: 4. Choose how many physical functions you need.
Physical Function *n*		
BAR0 Aperture BAR2 Aperture BAR4 Aperture	128 B - 256 GB	Default: 4 KB Choose the base address register aperture size.
BAR1 Aperture BAR3 Aperture BAR5 Aperture	128 B - 2 GB	Default: 4 KB Choose the base address register aperture size.
BAR0 Control BAR2 Control BAR4 Control	Disabled 32 bit I/O BAR 32 bit non-prefetchable memory BAR 32 bit prefetchable memory BAR 64 bit non prefetchable memory BAR 64 bit prefetchable memory BAR	BAR0 default: 32 bit non-prefetchable memory BAR BAR2, BAR4 default: Disabled Choose the BAR type.

Parameter	Choices	Notes
BAR1 Control BAR3 Control BAR5 Control	Disabled 32 bit I/O BAR 32 bit non-prefetchable memory BAR 32 bit prefetchable memory BAR	Default: Disabled Choose the BAR type.
Class Code	User defined	Default: 0x0
Device ID	User defined	Default: 0x100
Expansion ROM BAR Aperture	2 KB, 4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB, 16 MB, Disabled	Default: 4 KB
Interrupt Pin	NO INT, INTA, INTB, INTC, INTD	Default: NO INT
MSI Multiple Message Capable	1, 2, 4, 8, 16, 32	Choose the number of MSI messages the endpoint is allowed to send.
MSIX BAR Indicator	BAR0, BAR1, BAR2, BAR3, BAR4, BAR5	Default: BAR0
MSIX Capabilities Pointer	User defined	Default: 0xCo. Allowed values are 0x0 to 0xff.
MSIX Capability ID	User defined	Default: 0x11. Allowed values are 0x0 to 0xff.
MSIX PBA Indicator	BAR0, BAR1, BAR2, BAR3, BAR4, BAR5	Default: BAR0
MSIX PBA Offset	User defined	Default: 0x1. Allowed values are 0x0 to 0x1fffffff.
MSIX Table Offset	User defined	Default: 0x0. Allowed values are 0x0 to 0x1fffffff.
MSIX Table Size	User defined	Default: 0x0. Allowed values are 0x0 to 0x7ff.
Programming Interface Byte	User defined	Default: 0x0
Revision ID	User defined	Default: 0x0
Sub-Class Code	User defined	Default: 0x0
Subsystem ID	User defined	Default: 0x0

Table 162: Function Tab Settings (Root Port Mode)

Parameter	Choices	Notes
Class Code	User defined	Default: 0x0
Device ID	User defined	Default: 0x100
Programming Interface Byte	User defined	Default: 0x0
Revision ID	User defined	Default: 0x0
Sub-Class Code	User defined	Default: 0x0
Subsystem Vendor ID	User defined	Default: 0x1f7a
Vendor ID	User defined	Default: 0x1f7a

PCI Express Device Capability Tab

This tab defines the device capability settings.

Table 163: Reset Tab Settings

Parameter	Choices	Notes
Device Serial Number (DW1)	User defined	Default: 0x0. Allowed values are 0x0 to 0xffffffff.
Device Serial Number (DW2)	User defined	Default: 0x0. Allowed values are 0x0 to 0xffffffff.
Enable Slot Clock Configuration	On, off	Default: off
Extended Tag Field	On, off	Default: on
Link Port Number	User defined	Default: 0. Allowed values are 0 - 255.

PCI Express Pins Tab

This tab has several sub-tabs for defining the PCI Express pins. Refer to "Interface Signals" in the Titanium PCIe Controller User Guide for a complete definition of the pins and their function.

Table 164: Pins Tab: AXI Sub-Tab Settings

Parameter	Selection	Notes
Enable AXI Master Interface	On, off	Default: on. Indicate whether to use the AXI master.
Enable AXI Slave Interface	On, off	Default: on. Indicate whether to use the AXI slave.
AXI Clock Pin Name	User defined	Specify the pin name.
Invert AXI Clock Pin	On, off	Default: off. Indicate whether to invert the clock signal.
AXI Reset (Active-Low) Pin Name	User defined	Efinix recommends using the default names.
Master and slave tabs	User defined	These tabs define the pin names for the master and slave read/write data and address channels and sideband channel. Efinix recommends using the default names.

Table 165: Pins Tab: Interrupt Sub-Tab Settings

Parameter	Selection	Notes
Enable Interrupt	On, off	Default: off. Indicate whether to use the interrupt.
Interrupt Sideband Signals [27:0] Bus Name	User defined	Efinix recommends using the default names.
Local Error and Status Register Interrupt Pin name	User defined	Efinix recommends using the default names.
Legacy Interrupt tab	User defined	This tab defines the legacy interrupt pin names. Efinix recommends using the default names.
MSI tab	User defined	Available only when endpoint is selected. This tab defines the MSI pin names. To enable the MSI pins, turn on **Enable MSI**. Efinix recommends using the default names. MSI is not available in root port mode.

Table 166: Pins Tab: Message Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the PCIe message interface. Efinix recommends using the default names.

Table 167: Pins Tab: Error Indication Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the PCIe error interface. Efinix recommends using the default names.

Table 168: Pins Tab: APB Sub-Tab Settings

Parameter	Selection	Notes
APB Interface Clock Pin Name	User defined	Specify the pin name. The APB clock is required when using the PCI Express block for the PCIe Controller to enumerate successfully. The clock can be from an external source (through a GPIO) or from a PLL, If the clock comes from a PLL, the PLL must be configured in local feedback mode.
Invert APB Interface Clock Pin	On, off	Default: off. Indicate whether to invert the clock signal.
Pin and bus names	User defined	These fields define the APB interface pin names. Efinix recommends using the default names.

Table 169: Pins Tab: Function Level Reset Sub-Tab Settings

Parameter	Selection	Notes
All	User defined	Define the pin names for the PCIe FLR interface. Efinix recommends using the default names.

Table 170: Pins Tab: Status Sub-Tab Settings

Parameter	Selection	Notes
Enable Status	On, off	Default: on. Indicate whether to use the PCIe status signals.
Pin and bus names	User defined	These fields define the PCIe status pin names. Efinix recommends using the default names.

Table 171: Pins Tab: Configuration Snoop Sub-Tab Settings

Parameter	Selection	Notes
Enable Configuration Snoop	On, off	Default: on. Indicate whether to use the PCIe configuration snoop signals.
Pin and bus names	User defined	These fields define the configuration snoop pin names. Efinix recommends using the default names.

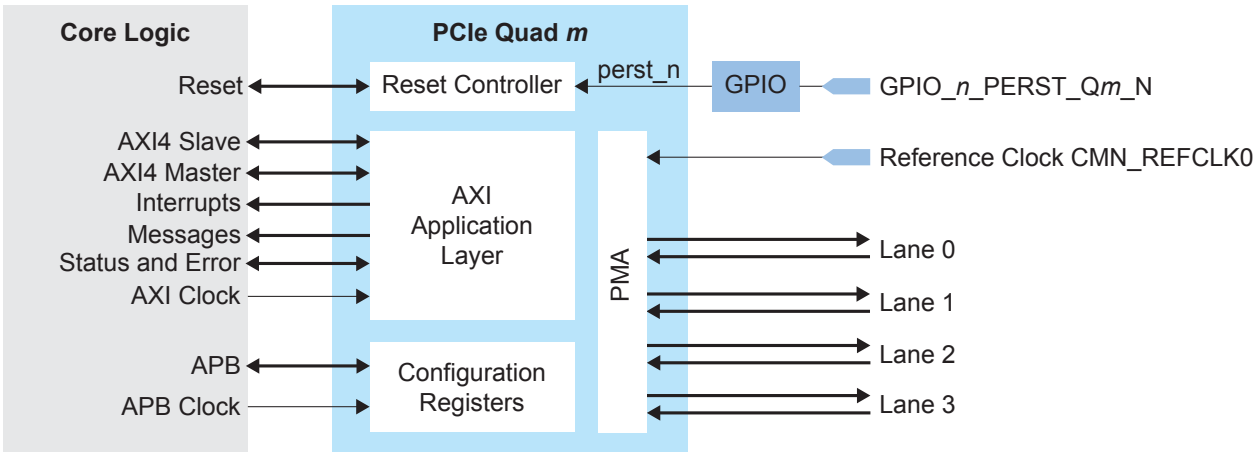
Table 172: Pins Tab: Power Management Sub-Tab Settings

Parameter	Selection	Notes
Enable Power Management	On, off	Default: off. Indicate whether to use power management.
ASPM L1.1 Substate Enable	On, off	In root port mode, choosing **L0s and L1 Entry** or **L1 Entry** activates these options. Default: on
ASPM L1.2 Substate Enable	On, off	
PM L1.1 Substate Enable	On, off	Default: off. Indicate whether to use PM L1.1 substate.
PM L1.2 Substate Enable	On, off	Default: off. Indicate whether to use the PM L1.2 substate.
ASPM Enable	Disabled L0s Entry L0s and L1 Entry L1 Entry	Choose the type of active state power management (ASPM) to use.
Power Management Clock Pin Name	User defined	Specify the pin name.
Power Management Clock Connection Type	rclk, gclk	Choose whether to use a global clock (gclk) or regional clock (rclk) for the power management clock pin.
Pin and bus names	User defined	These fields define the power management pin names. Efinix recommends using the default names.
L1 Substate tab pin and bus names	User defined	These pins are used when the ASPM L1.1, ASPM L1.2, PM L1.1, or PM L1.2 substates are enabled. Efinix recommends using the default names.

Using the PCI Express Interface

You use the Efinity Interface Designer to implement a PCI Express block and configure it. Additionally, your interface design requires a PLL and a GPIO block for clock and reset signals. The PERST pin is a GPIO resource. The reference clock pin is a dedicated package pin.

Figure 70: Interface Designer Block Diagram



Where *n* is the GPIO resource number and *m* is the quad number.

See **GPIO Block (PERST_N)** on page 234 and **Reference Clock** on page 234 for the resource and pin information.

Note: You also need to enable the configuration user status pin. See **Enable Configuration User Status Pin** on page 235.

Additionally, the PCI Express block requires you to have an APB clock. Specify a clock name in the Interface Designer.

Tip: The PCIe Express interface is complicated. If you find yourself turning a lot of options on and off to enable or disable features, some of the GUI menus might get out of sync. Therefore, Efinix suggests that you note the settings you want, and then start fresh with a newly created PCI Express block with only the settings you want.

GPIO Block (PERST_N)

For the `PERST_N` signal, instantiate a GPIO block as an input with these settings:

Table 173: GPIO Block Settings for PERST_N

Parameter	Setting
Pin Name	gpio_PERSTN0
Connection Type	pcie_perstn
Pull Option	weak pullup
Static Delay Setting	0

In the Resource Assigner, assign the resource shown in the **PCI Express > Block Editor > Reset tab** for the PCIe resource you selected.

Table 174: PERST_N Resource

FPGA	Transceiver Quad	Resource
Ti165, Ti240, Ti375	QUAD_0	GPIOR_141
	QUAD_2	GPIOR_144
Ti85, Ti135	QUAD_0	GPIOR_131

Reference Clock

The PCIe interface reference clock is a dedicated package pin.

Table 175: Reference Clock Pin

FPGA	Transceiver Quad	Reference Clock
Ti165, Ti240, Ti375	QUAD_0	Q0_REFCLK0_P/N
	QUAD_2	Q2_REFCLK0_P/N
Ti85, Ti135	QUAD_0	Q0_REFCLK0_P/N

Oscillator

The transceiver is connected directly to the high-performance oscillator. If you want to use the oscillator for other logic as well as the transceiver, set up the oscillator to be free running.

1. Add the Oscillator block to your interface design.
2. Leave **Enable Oscillator Pin Name** empty. Leaving it empty makes the oscillator free running.

Enable Configuration User Status Pin

You may want your PCIe application to know when the link is ready for TLP transactions. Your application can monitor the FPGA's configuration user status pin. This signal is generated by the FPGA's configuration control block. When the FPGA asserts this status signal, your application can start TLP transactions.

Important: When you are using the PCI Express block, you must enable the configuration user status pin. Otherwise the Interface Designer issues an error. See configuration_rule_in_user_pin.

To enable this signal in the Interface Designer:

1. Go to **Device Setting > Configuration > User Status tab**.
2. Turn on **Enable User Status Control**. The **User Status Pin Name** shows `cfg_USR_STATUS`. This status pin can drive your PCIe application.

Learn more: Refer to "Link Up" in the Titanium PCIe Controller User Guide for a timing diagram showing the `cfg_USR_STATUS` signal during link up.

Design Check: PCI Express Messages

When you check your design, the Interface Designer applies design rules to your configuration settings. The following tables show some of the error messages you may encounter and explains how to fix them.

pcie_rule_apb_clock (error)

Message	PLL <*PLL instance name*> with output clock driving APB clock must have its reference clock mode set to external
To fix	In the PLL Clock Calculator, set PLL reference clock mode to external for the PLL instance in the message.

Message	PLL <*PLL instance name*> with output clock driving APB clock must be configured as local feedback mode
To fix	In the PLL Clock Calculator, set PLL reference clock mode to local feedback for the PLL instance in the message.

Message	APB Clock from PLL frequency <#>MHz is out of range. Max=200MHz
To fix	Change the PLL frequency to be lower than the specified limit.

Message	APB Clock pin cannot be empty
To fix	Enter the APB clock pin name (**Block Editor** > **Pins tab** > **APB sub-tab** > **APB Interface Clock Pin Name**).

Message	APB enable is compulsory
To fix	You may receive this error if you are migrating your design between software versions. This error causes an internal error. To fix it, enter the APB clock pin name (**Block Editor** > **Pins tab** > **APB sub-tab** > **APB Interface Clock Pin Name**).

pcie_rule_axi (error)

Message	At least either AXI Master or AXI Slave has to be enabled
To fix	Enable a master or slave interface (**Block Editor** > **Pins tab** > **AXI sub-tab**).

pcie_rule_axi_clock (error)

Message	PCIe AXI Clock from PLL frequency <#>MHx is out of range. Min=<#>MHz Max=<#>MHz
To fix	Change the PLL frequency to a valid range. The expected range is 125 to 250 MHz.

Message	AXI Clock pin name has not been configured
To fix	Enter the AXI clock pin name (**Block Editor** > **Pins tab** > **AXI sub-tab** > **AXI Clock Pin Name**).

pcie_rule_external_clock (error)

Message	PCIE only allows external clock 0 to be configured.
To fix	In the *<project>*.**peri.xml** file, update the value of `PMA_CMN__cmn_plllc_gen_preg__cmn_plllc_pfdclk1_sel_preg` to `Refclk 0`.

pcie_rule_hw_drc (error)

Message	Invalid value assigned to the following parameters: <list_of_parameters> Found <#> HW Errors: <list_of_errors>
To fix	Enter valid values for the parameters.

pcie_rule_hw_drc (warning)

Message	Found <#> HW Warnings: <list_of_warnings>
To fix	Enter valid values for the parameters.

pcie_rule_inst_name (error)

Message	Instance name is empty. Valid characters are alphanumeric characters with dash and underscore only
To fix	Enter a valid pin name.

pcie_rule_invalid_hex_value (error)

Message	The following hexadecimal parameters has invalid value: <list_of_parameters_with_error_message>
To fix	Update the parameters to use a hexadecimal value in a valid range.

pcie_rule_osc_clock (error)

Message	Oscillator is required to be configured
To fix	Create an Oscillator instance with no enable pin.

pcie_rule_perstn (error)

Message	PERSTN resource <resource_name> is not configured as perstn connection
To fix	Set the connection type for the GPIO instance with resource <resource_name> to pcie_perstn. See **GPIO Block (PERST_N)** on page 234.

Message	PERSTN resource <resource_name> input name is empty
To fix	Enter an input name for the GPIO instance with resource <resource_name>.

Message	PERSTN resource <resource_name> is not configured as input
To fix	Set the mode to input for GPIO instance with resource <resource_name>.

Message	The PERSTN resource <resource_name> has not been configured
To fix	Create a GPIO instance with resource <resource_name>.

pcie_rule_pm_clock (error)

Message	Empty power management clock pin name
To fix	Enter the power management clock pin name or disable power management (**Block Editor** > **Pins tab** > **Power Management sub-tab**).

pcie_rule_refclk (error)

Message	Invalid <external_reference_clock> selection due to pins not available in device
To fix	Assign external reference clock resource that exists in the FPGA. Check the spelling of the resource name if you are using the Python API.

Message	PLL instance with resource <PLL resource list> is required to be configured when reference clock is not from on-board crystal
To fix	Create a PLL instance and assign one of the resources listed in the message, or turn on the **PCI Express block** > **Base tab** > **Reference clock from on-board crystal** option.

Message	Either one of the following PLL instance(s) required to enable output clock when reference clock is not from on-board crystal: <list of PLL instances with expected output clock>
To fix	Enable the output clock for one of the named PLL instances.

Message	Either one of the following PLL instance(s) required to use <list of reference clock source> as exteranl reference clock source when PCIe reference clock is not from on-board crystal: <list of PLL instances>
To fix	For one of the PLL instances, set external reference clock and use the listed resource as the reference clock source.

Message	Either one of the following PLL instance(s) required to use local feedback mode when reference clock is not from on-board crystal: <list of PLL instances>
To fix	Change the PLL's feedback mode to local for one of the named PLL instances.

Message	Core refclk pin has to be specified in core mode
To fix	Enter a reference clock pin name (Q*n*_REFCLK0/1_N/P). Check the spelling of the resource name if you are using the Python API.

Message	Reference clock PLL resource <PLL_resource_name> Output Clock <#> has not been configured
To fix	Create a PLL instance with resource <PLL_resource_name> and enable output clock <#>.

Message	Reference clock PLL resource <PLL_resource_name> has not been configured
To fix	Create PLL instance with resource <PLL_resource_name>.

Message	Invalid reference clock since PLL does not exists in the device
To fix	Choose another connection type.

pcie_rule_resource (error)

Message	Resource name is empty
To fix	Assign a resource.

Message	Resource is not a valid PCIe device instance
To fix	Assign the instance to a resource that exists in the FPGA. Check the spelling of the resource name if you are using the Python API.

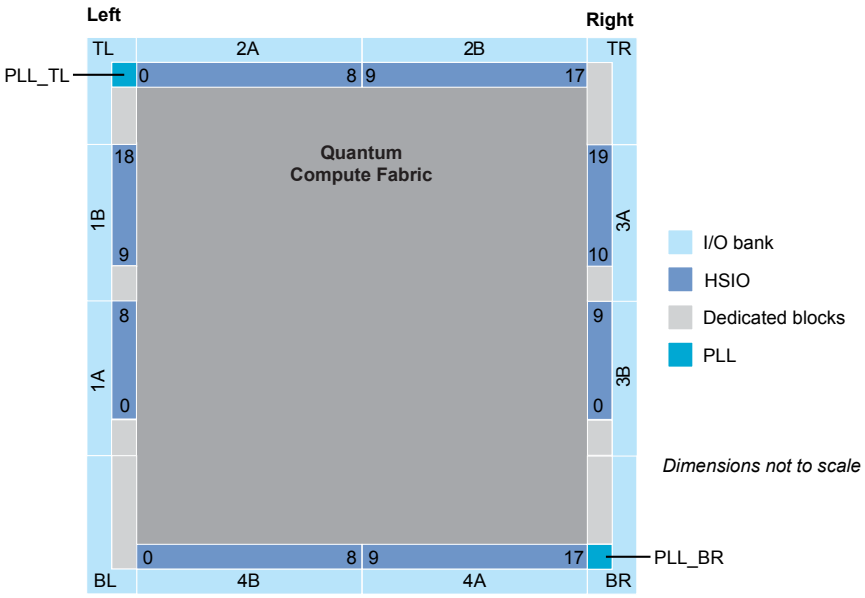
Chapter 20

Interface Floorplans

Note: The numbers in the floorplan figures indicate the HVIO and HSIO number ranges. Some packages may not have all HVIO or HSIO pins in the range bonded out. Refer to the pinout for information on which pins are available in each package.

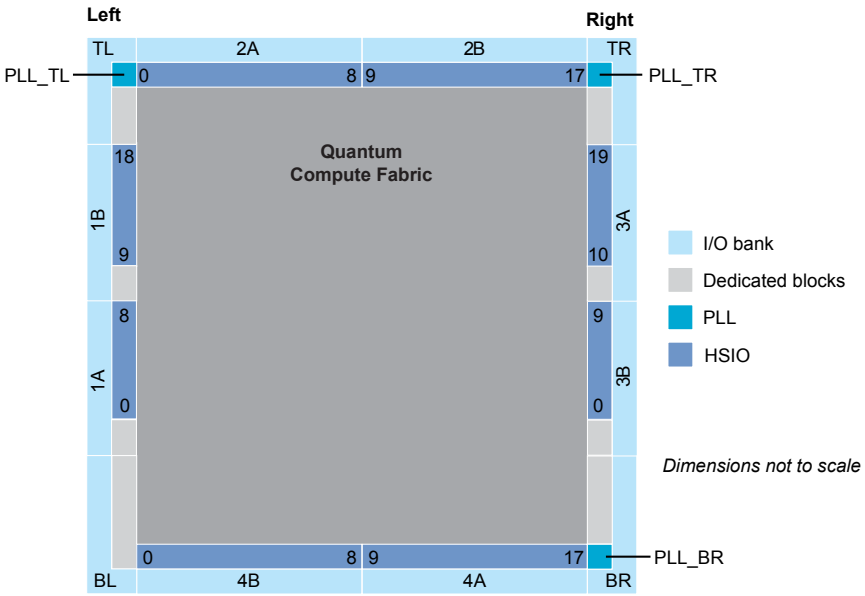
Floorplan Diagram for FPGAs in W64 Packages

Figure 71: Ti60 FPGAs



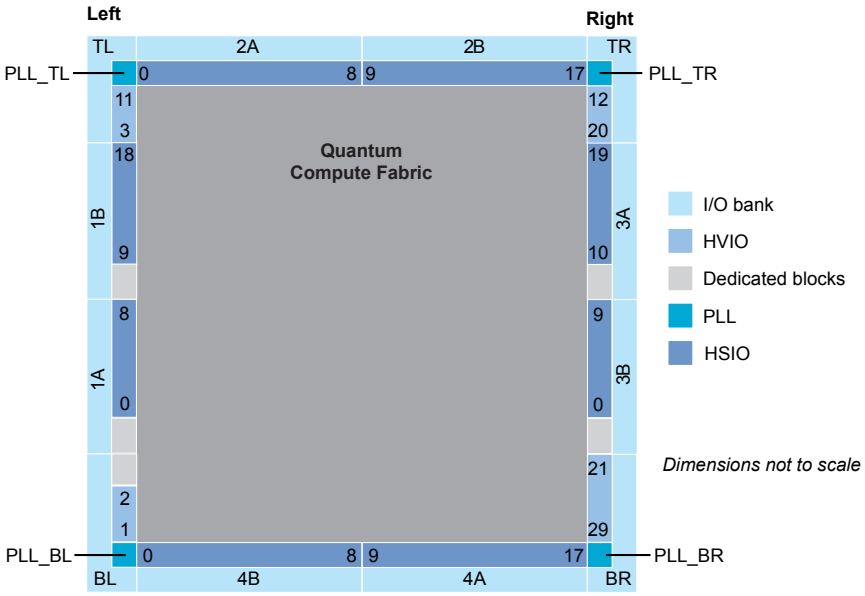
Floorplan Diagram for FPGAs in F100 and F100S3F2 Packages

Figure 72: Ti35 and Ti60 FPGAs



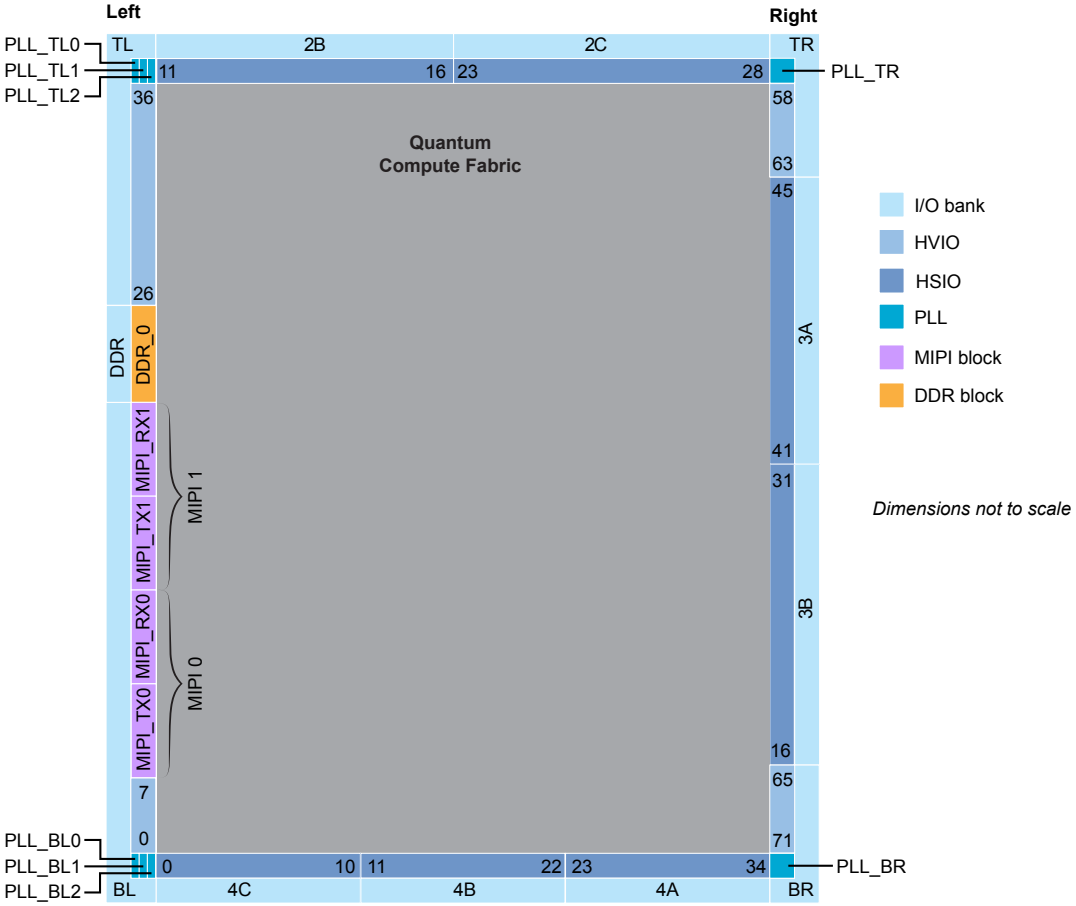
Floorplan Diagram for FPGAs in F225 and F256 Packages

Figure 73: Ti35 and Ti60 FPGAs



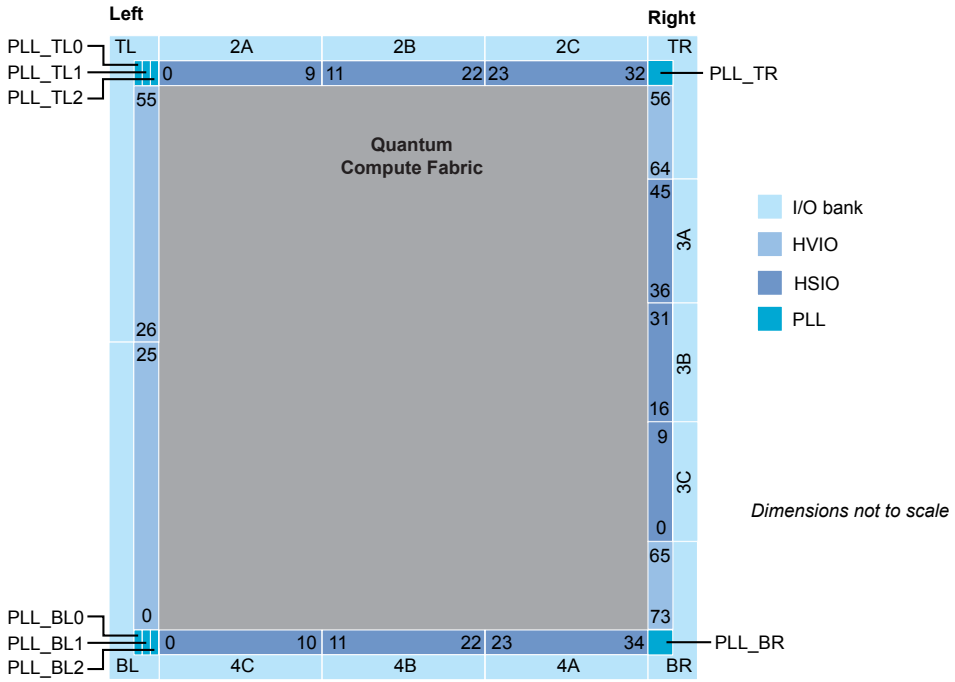
Floorplan Diagram for FPGAs in J361 Packages

Figure 74: Ti90, Ti120, and Ti180 FPGAs



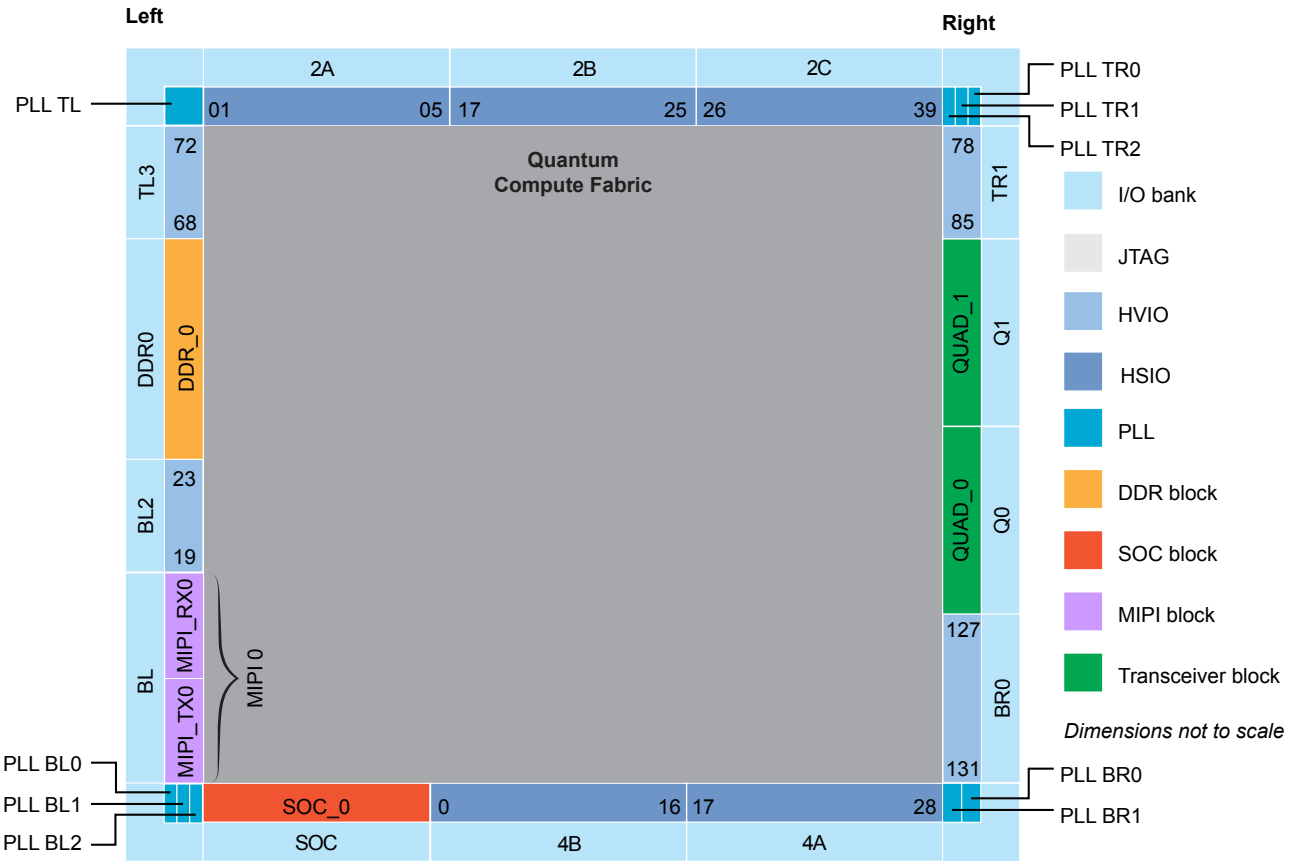
Floorplan Diagram for FPGAs in G400 Packages

Figure 75: Ti90, Ti120, and Ti180 FPGAs



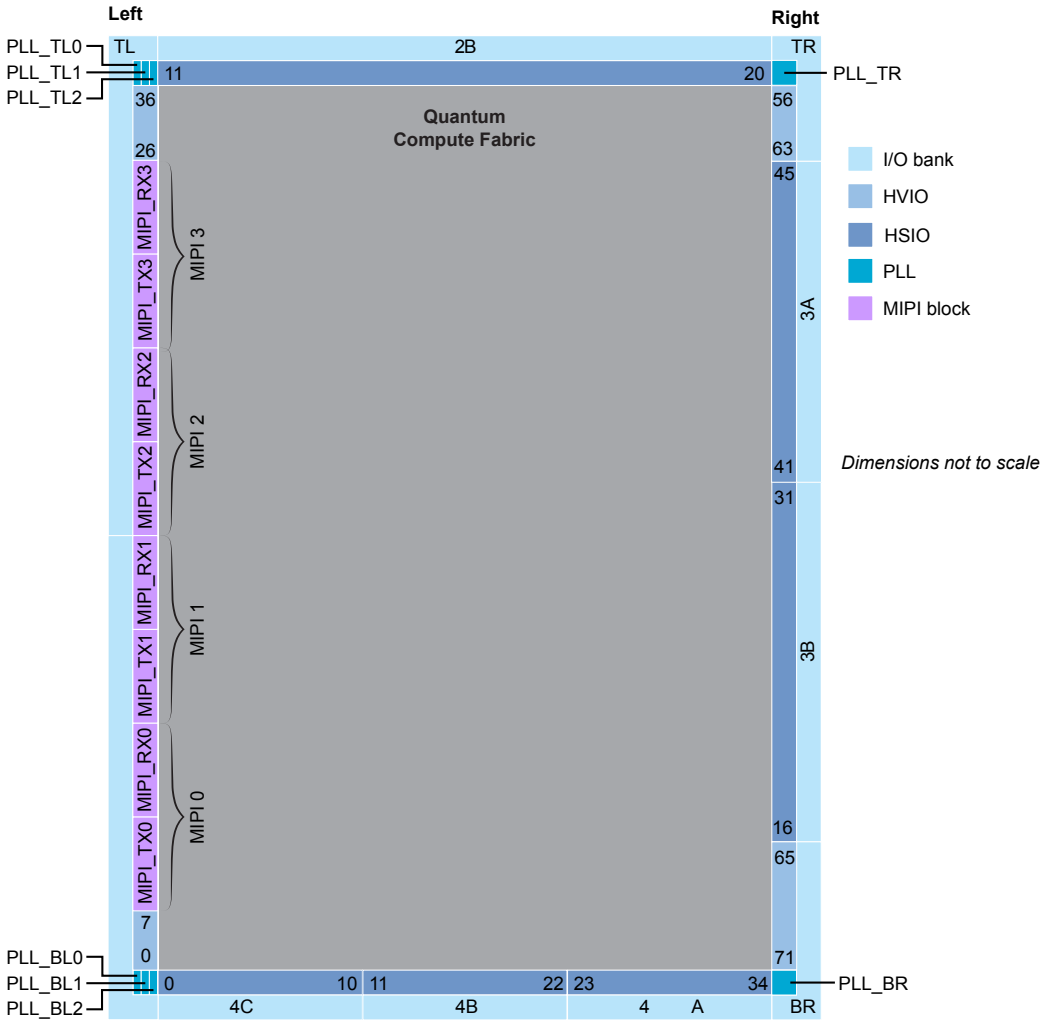
Floorplan Diagram for FPGAs in N441 Packages

Figure 76: Ti85 and Ti135 FPGAs



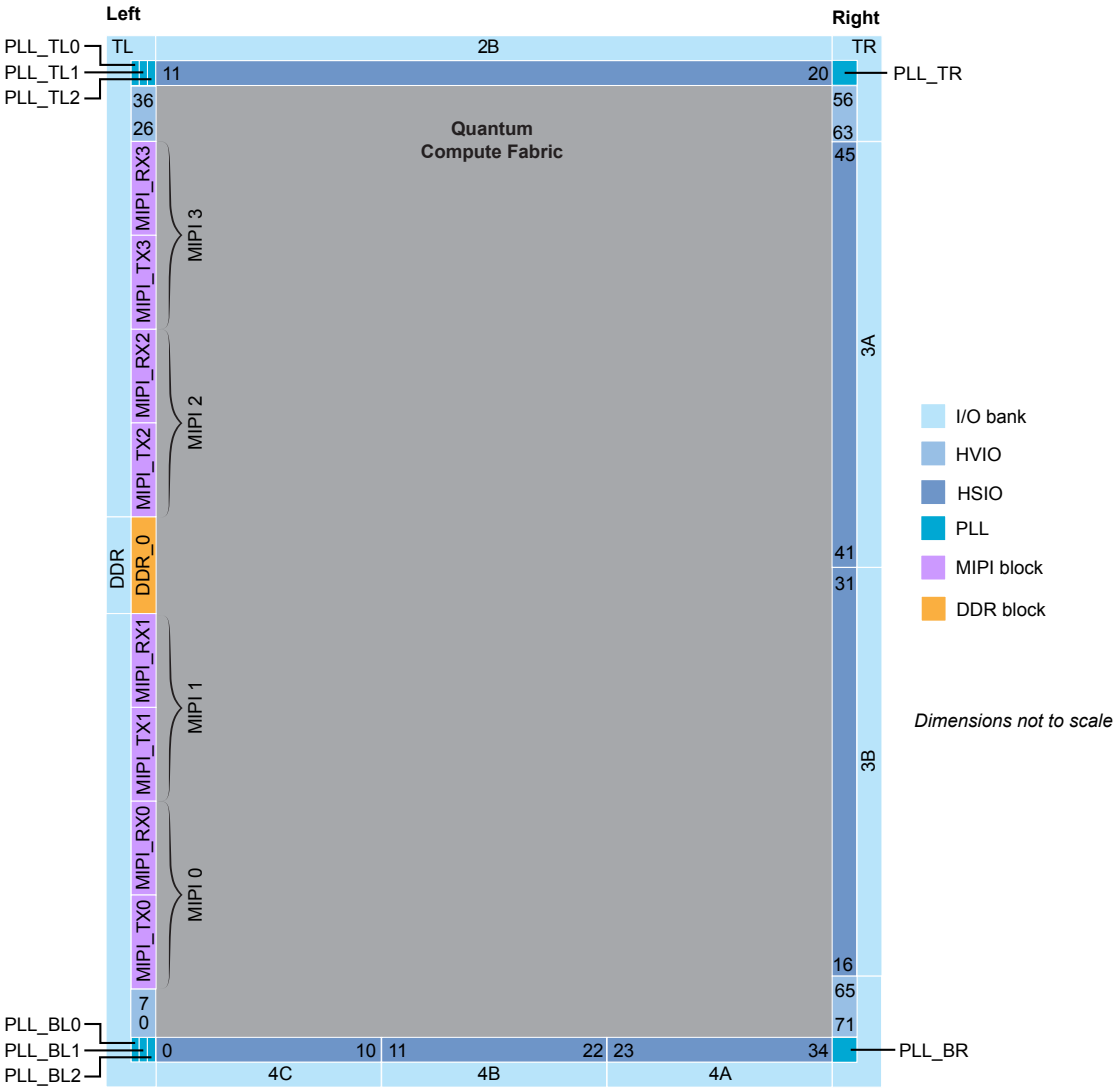
Floorplan Diagram for FPGAs in L484 Packages

Figure 77: Ti90, Ti120, and Ti180 FPGAs



Floorplan Diagram for FPGAs in J484 and M484 Packages

Figure 78: Ti90, Ti120, and Ti180 FPGAs

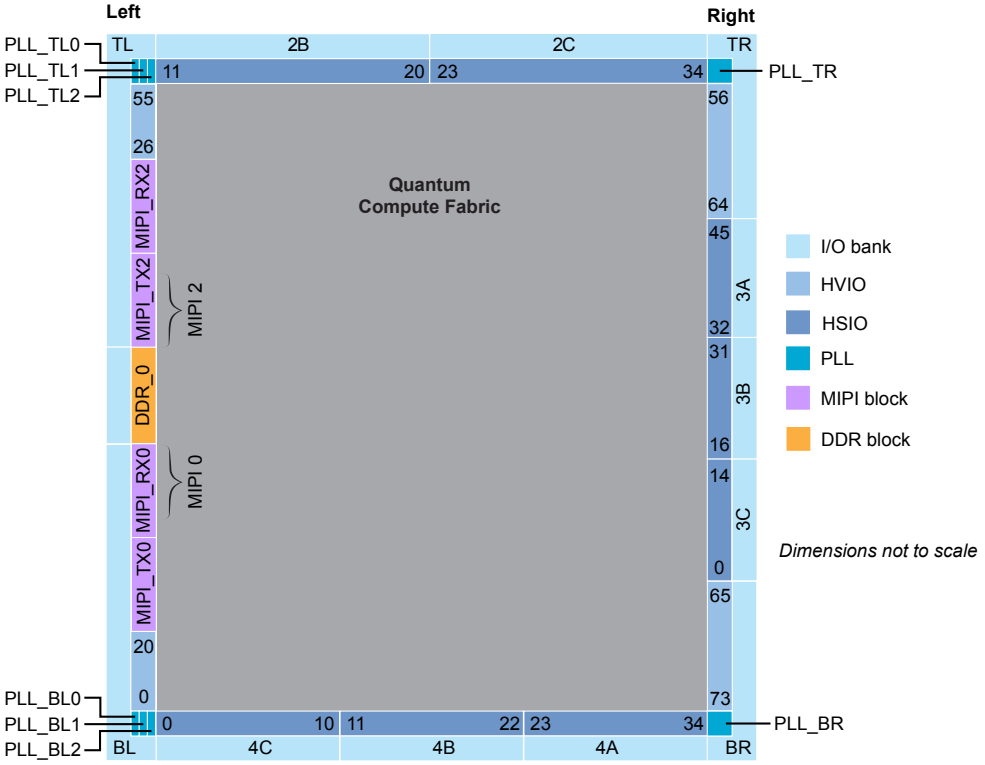


Ti90 and Ti120 FPGAs are available in the J484 package only.

Ti180 FPGAs are available in the J484 and M484 packages.

Floorplan Diagram for FPGAs in J484D1 Packages

Figure 79: Ti180 FPGAs



Floorplan Diagram for FPGAs in N484 Packages

Figure 80: Ti165, Ti240, and Ti375 FPGAs

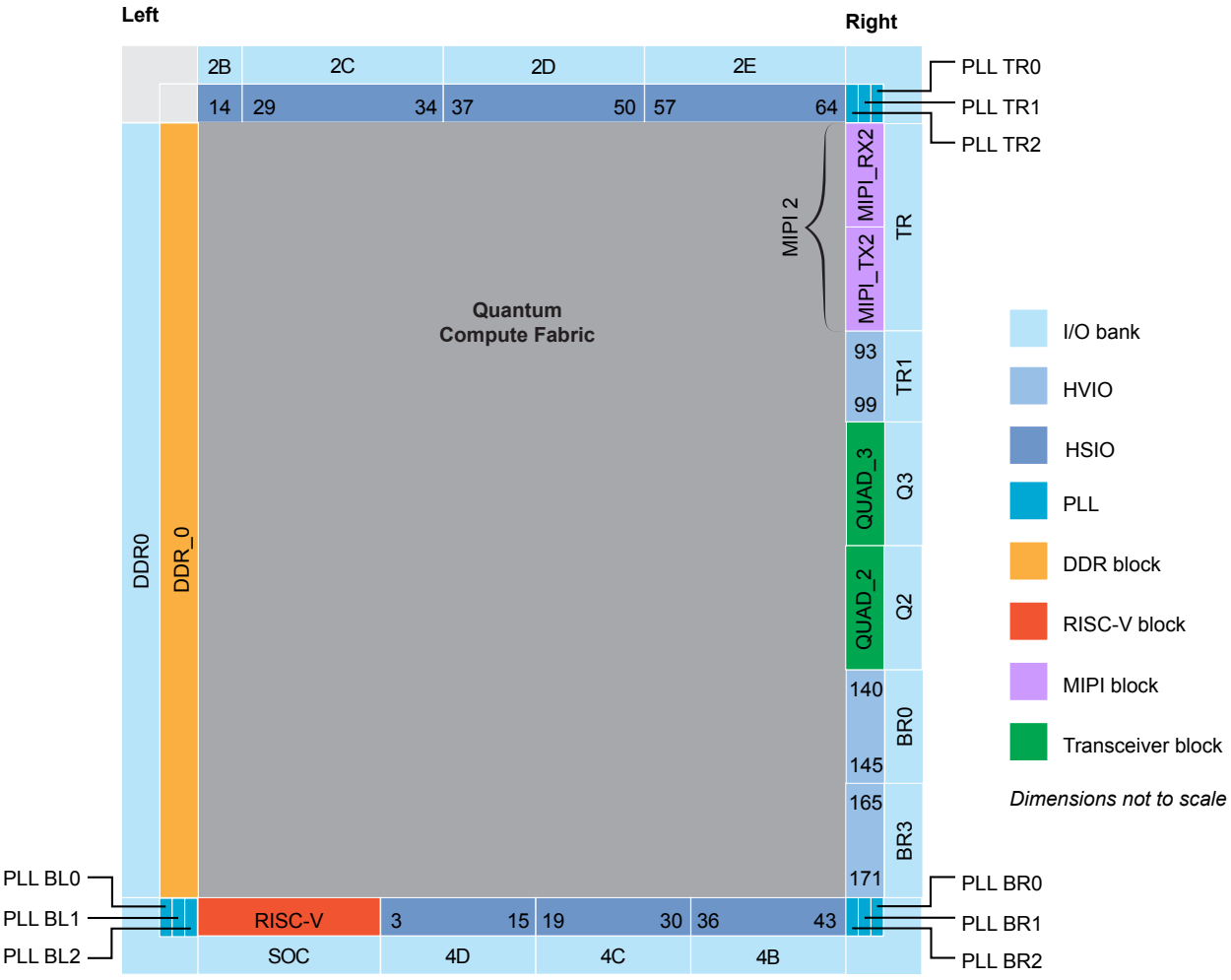
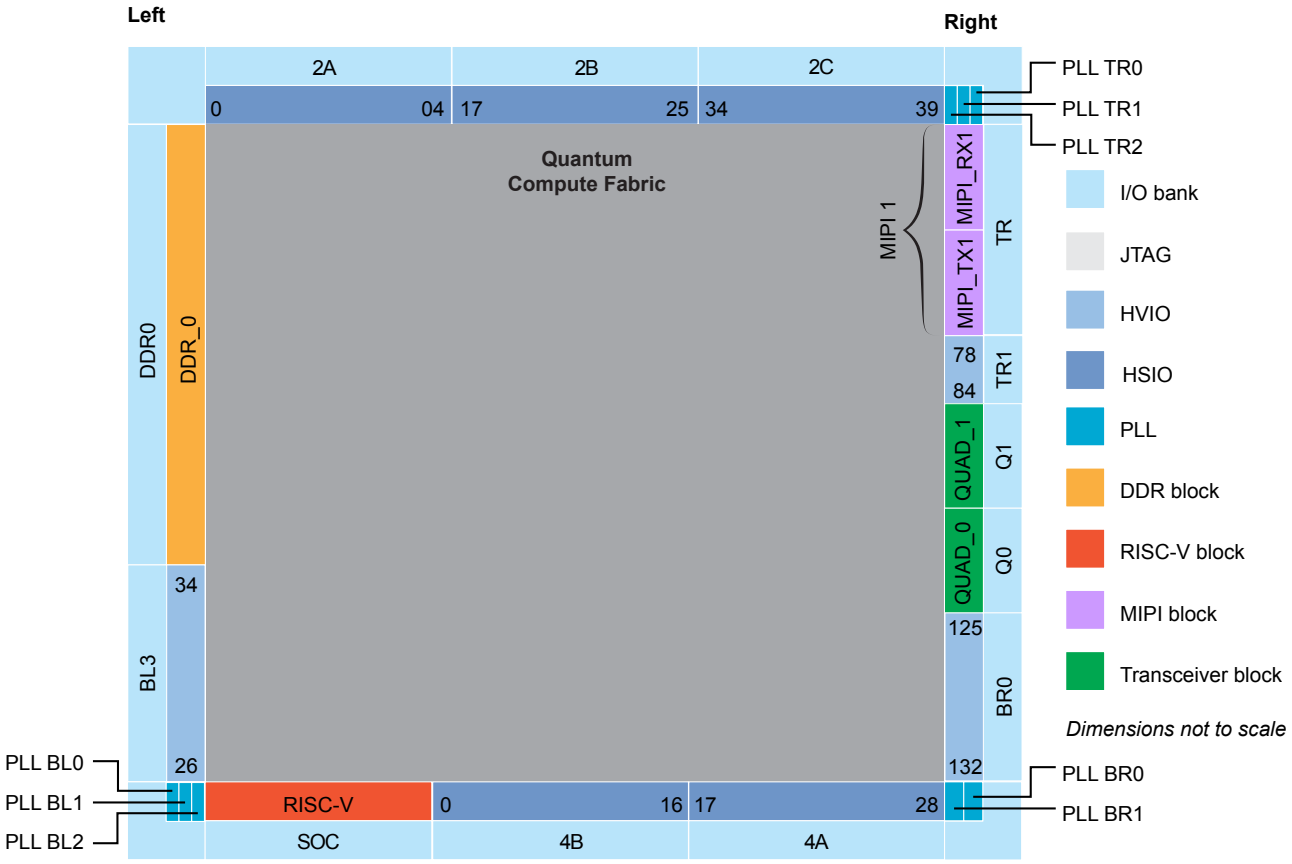
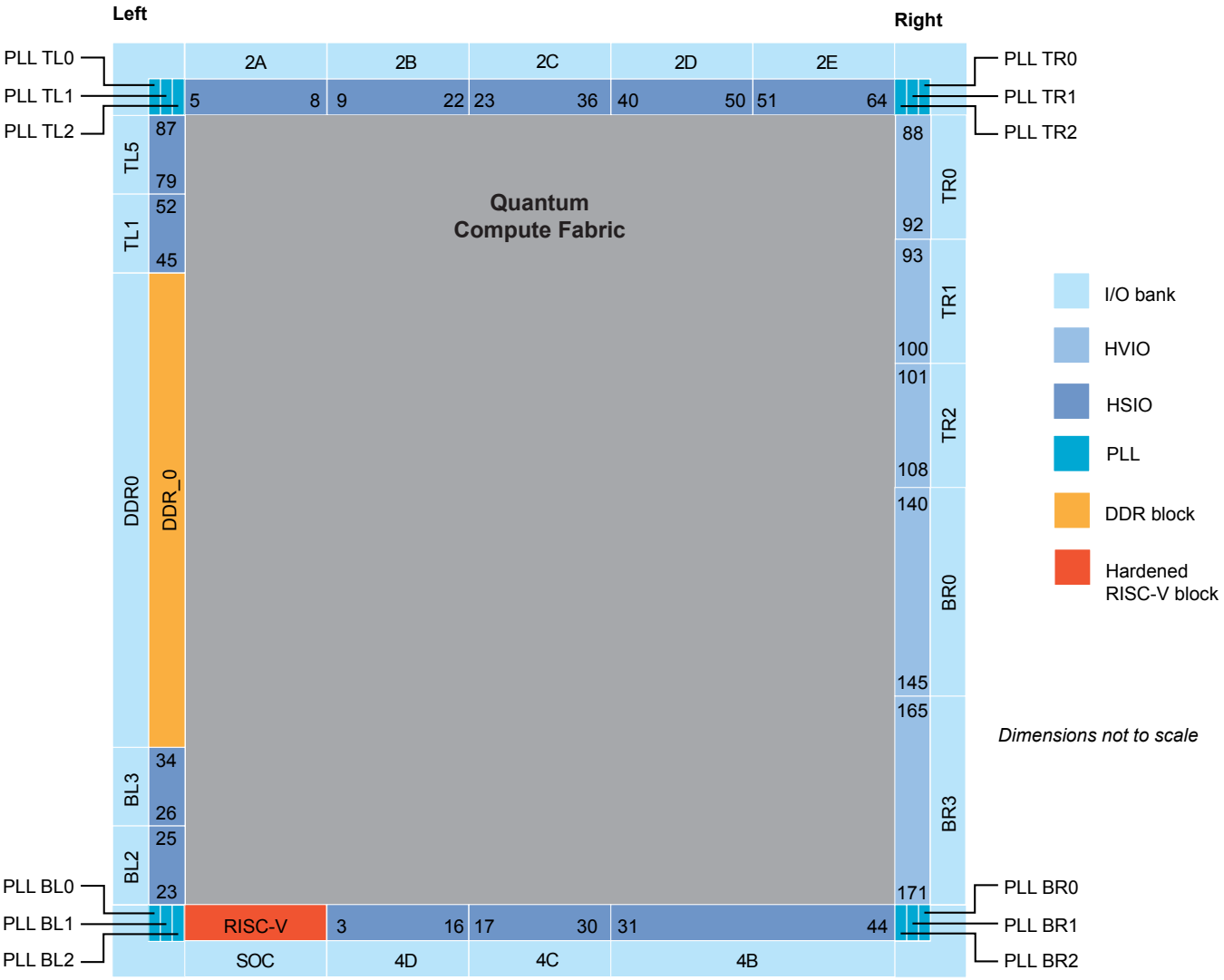


Figure 81: Ti85 and Ti135 FPGAs



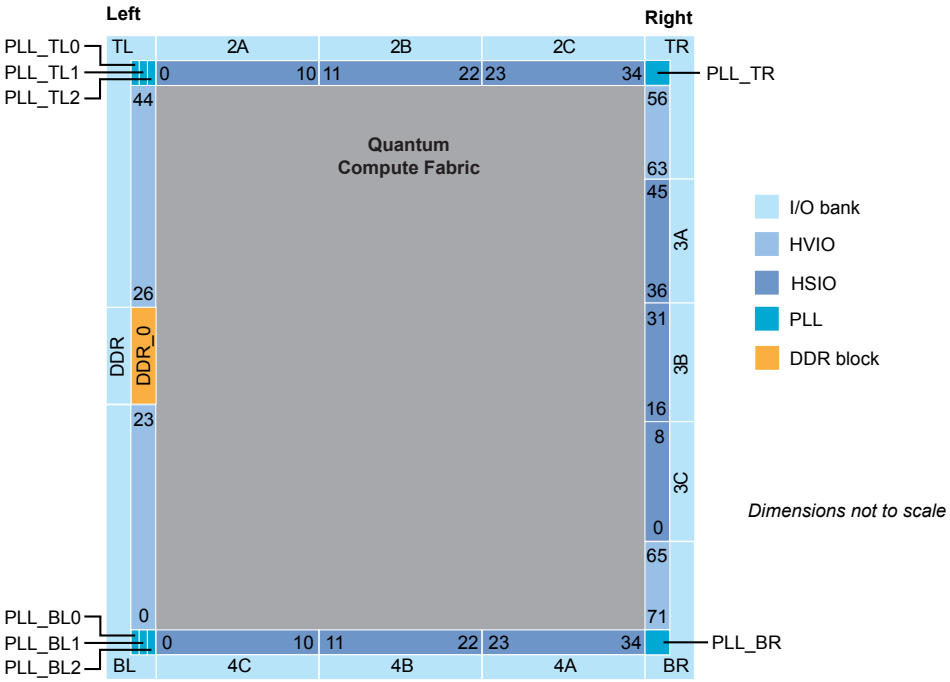
Floorplan Diagram for FPGAs in C529 Packages

Figure 82: Ti165, Ti240, and Ti375 FPGAs



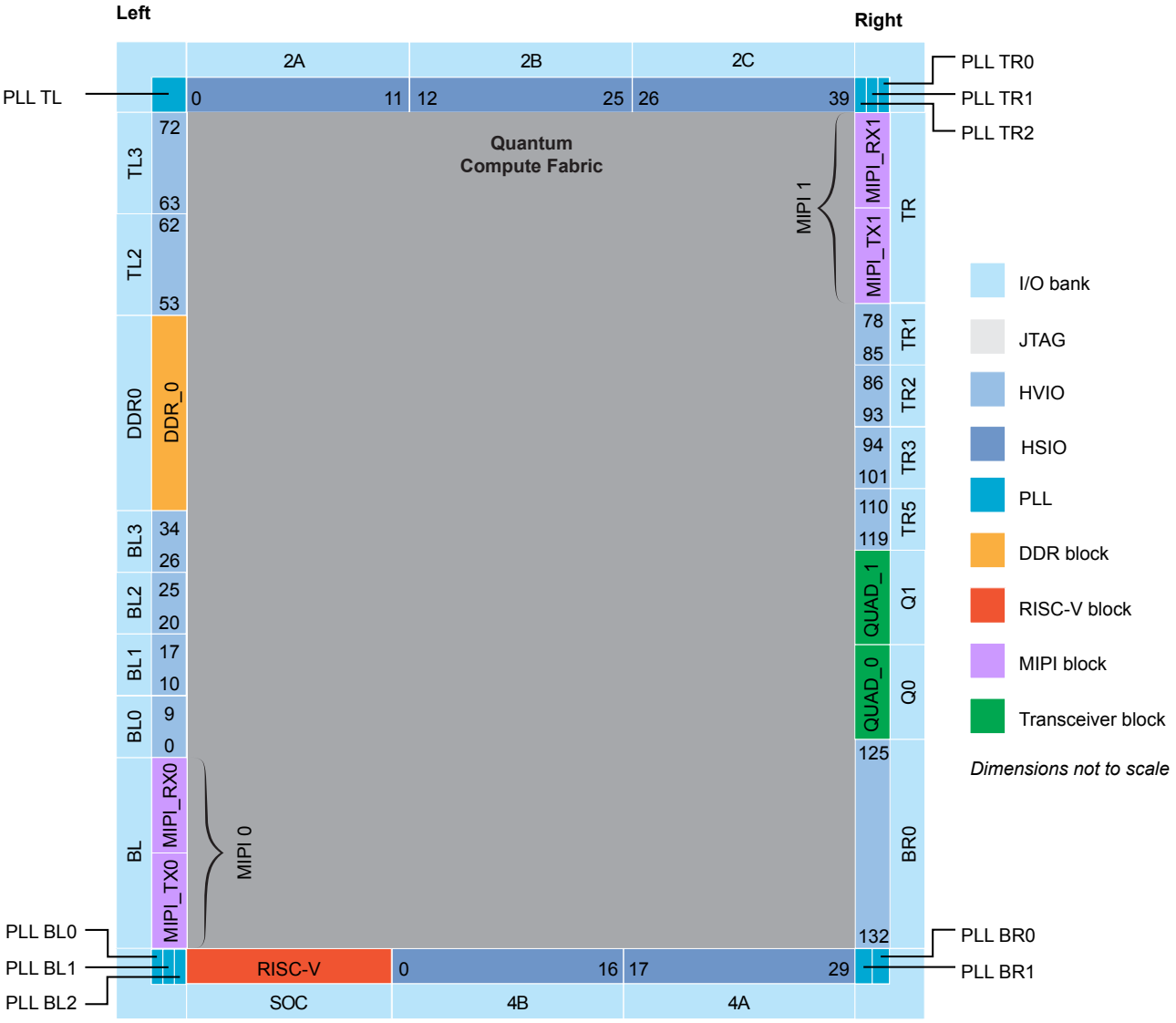
Floorplan Diagram for FPGAs in G529 Packages

Figure 83: Ti90, Ti120, and Ti180 FPGAs



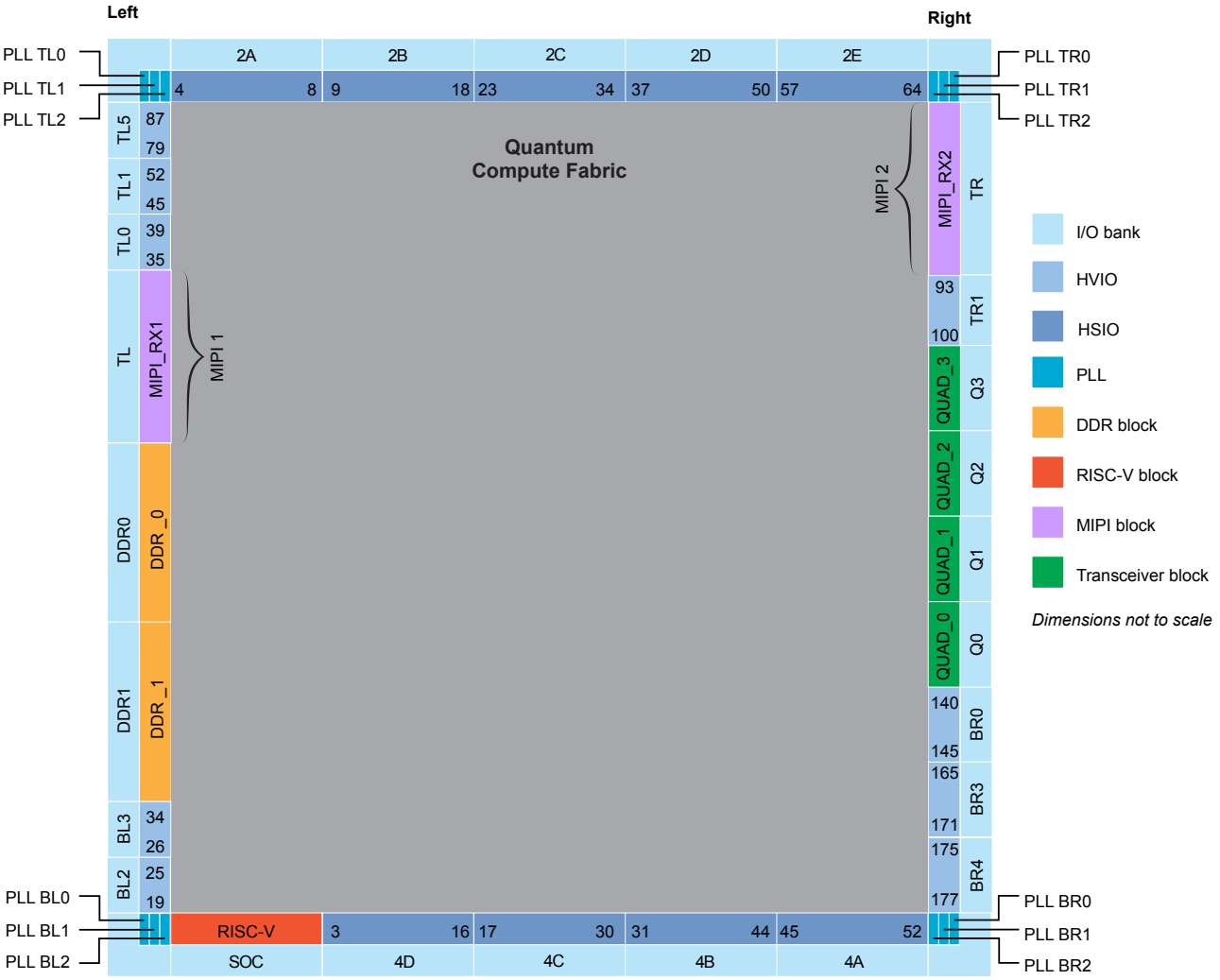
Floorplan Diagram for FPGAs in N676 Packages

Figure 84: Ti85 and Ti135 FPGAs



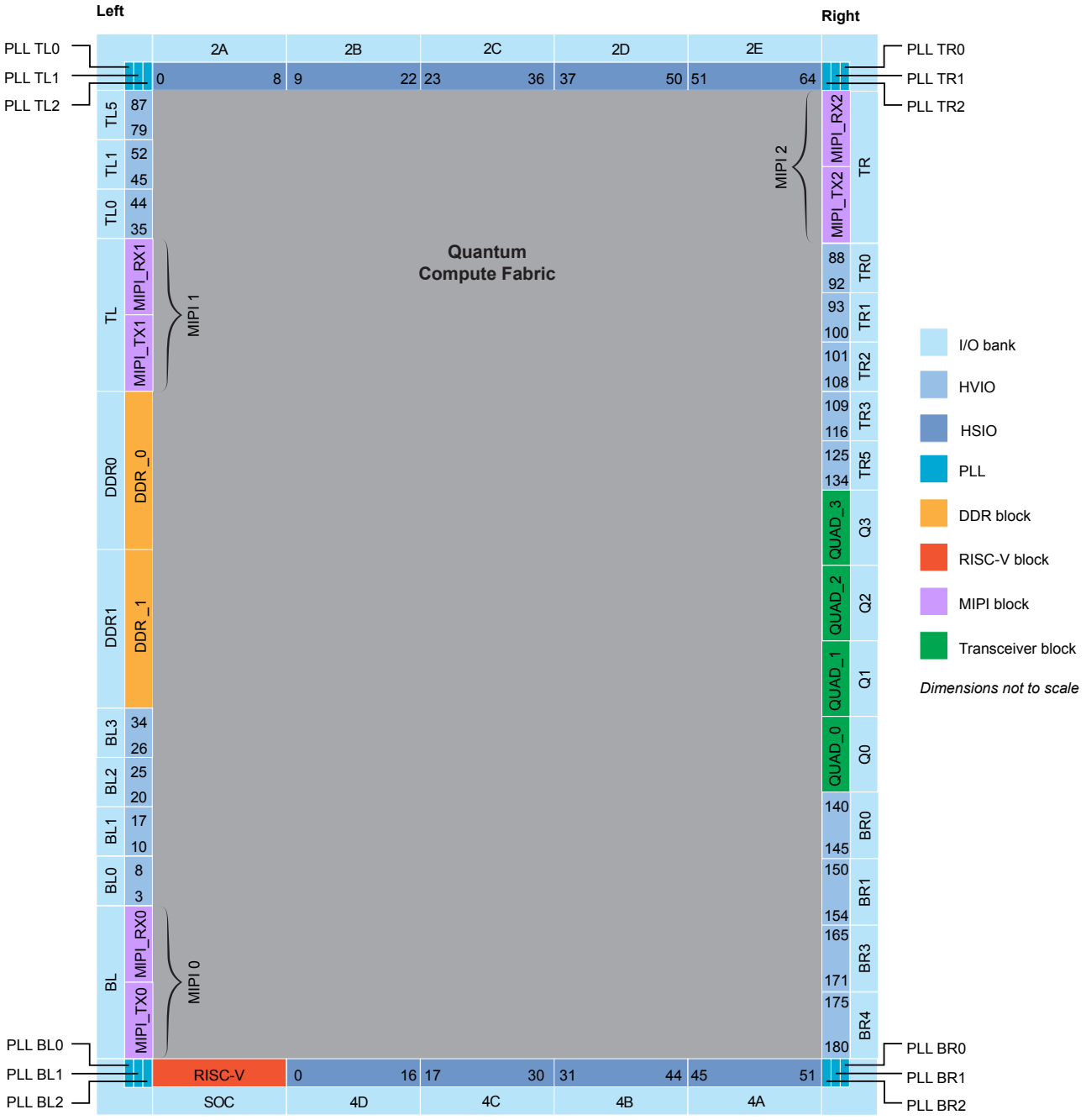
Floorplan Diagram for FPGAs in N900 Packages

Figure 85: Ti165, Ti240, and Ti375 FPGAs



Floorplan Diagram for FPGAs in N1156 Packages

Figure 86: Ti165, Ti240, and Ti375 FPGAs



Icon Reference

Interface Designer Icons


Interface Designer


Add Block

Create a GPIO bus


Delete Block


Show or Hide Block Editor


Resource Assigner

Export GPIO Assignments


Import GPIO Assignments


Clear Design


Check Design for Errors


Export Settings


Generate Constraints File


Package Planner

Resource Assigner


Toggle Instance View and Resource View


Clear Resource

Clear All Resources


Show/Hide Filter


Clear Filter

Revision History

Table 176: Revision History

Date	Version	Description
June 2025	6.1	Added note that user should perform IBIS simulation to determine which settings to use for the DDR block (VREF and DQ settings). (DOC-2432) When enabling the HSIO dynamic delay, the delay is updated on the rising edge of DLYCLK. (DOC-2531) Fix typos. (DOC-2537) The HyperRAM block has five new options to set the drive strength of the clock, reset, chip select, data strobe, and bus signals. (DOC-2570) Added Design Check message for HyperRAM drive strength check. (DOC-2570)
May 2025	6.0	Updated interface floorplan diagrams. Updated **Package/Interface Support Matrix** on page 10. Added **Enable SLVS** option to **Table 49: LVDS RX Options** on page 97. Added new warning messages to **Design Check: RISC-V Messages** on page 187. (DOC-2252) Updated messages for clock and control, LVDS (DOC-2453), PLL (DOC-2335), and oscillator. Corrected choice range for PLL clock divider. (DOC-2524) Ethernet XGMII now supports 5 Gbps speed setting. (DOC-2326) Updated messages for transceivers (DOC-2503).
January 2025	5.6	Added Ethernet SGMII interface. Renamed 10GBase-KR interface as Ethernet XGMII. (DOC-2294) Updated PMA Direct presets. (DOC-2270) Updated PCI Express settings. Added pma_direct_rule_data_rate_timing_model design check rule. (DOC-2270) The clock_rule_lvds_rx_clock_source message is a warning not an error. (DOC-2308) Ti85 and Ti135 are available in N484 packages.
December 2024	5.5	Ti165, Ti240, and Ti375 FPGAs in N484 packages now support PCIe Gen4. (DOC-2240) The PCI Express block requires that you have a valid APB clock. (DOC-2167) Added new option for PCI Express block (Base tab) about whether to use a reference clock from an on-board crystal. (DOC-2222) Updated PCI Express messages. (DOC-2222 and DOC-2264) Added **PMA Direct block** > **Common Properties tab** > **Clock and Reset sub-tab** and new options. (DOC-2224) Updated PMA Direct messages for new reset pin name rule. (DOC-2224) Added more preset options for PMA Direct block. (DOC-2237) Changed the pll_rule_feedback_clock message about 0 degree phases for feedback clocks from a warning to an info message. (DOC-2252) Updated interface diagrams to align with resource naming in Efinity software.

Date	Version	Description
November 2024	5.4	Removed Enable Advanced Peripheral Bus option for PCIe. This option must always be turned on. (DOC-2174) clock_rule_undefined_name is now an info message not a warning (**Design Check: Clock Control Messages** on page 41) (DOC-2071) Added warning message for pll_rule_feedback_clock about 0 degreed phases for feedback clocks. (DOC-2036) Removed PLL IOFBK interface pin. For 10Gbase-KR, changed GUI name for auto-negotiation to **Enable Auto Negotiation (AN) Clause 37**. (DOC-2194) When you enable dynamic delay, the delay is updated on the falling clock edge of DLYCLK. (DOC-2159) Updated GPIO and LVDS interface pin names (IN to I and OUT to O) to align with primitives. (DOC-2086) Corrected steps for **Create a MIPI TX Interface** on page 126. (DOC-2157) Updated MIPI Lane messages. (DOC-2157) Added note to create new interface block for PCI Express, 10GBase-KR, and PMA Direct if you have made many option changes. (DOC-2067)
October 2024	5.3	Correct typos.
October 2024	5.2	Added **PMA Direct Interface** on page 209 chapter. Added topic about how to **Enable External Access to Flash** on page 24. Added note that the **Enable User Status Pin** option is only available for FPGAs with transceivers. (DOC-2085) Added common_quad_lane_rule_protocol message for 10GBase-KR interface.
August 2024	5.1	Added Ti85 and Ti135 FPGAs. **Enable KR Base** option moved to **Pins** > **Control** tab. (DOC-2026) Added floorplan diagram for N676 package.
June 2024	5.0	Added **PCI Express Interface** on page 226 and **Ethernet XGMII Interface** on page 202 chapters. (DOC-1808) Updated table of interface blocks and package/interface support matrix. Updated **Design Check: SEU Messages** on page 27 messages. Updated **Design Check: Configuration Messages** on page 28. Added N484 and N1156 interface floorplan figures. (DOC-1808) Updated C529 interface floorplan figure. Added reset recommendations for PLLs and cascaded PLLs. (DOC-1900) Clarified which signals are available when LVDS settings are enabled. (DOC-1908)
May 2024	4.4	Added Ti165 and Ti240 FPGAs, replacing the Ti135 and Ti240, respectively.
March 2024	4.3	Removed M361, M484, and F529 packages for Ti90 and Ti120 FPGAs. Removed M361 and F529 packages for Ti180 FPGAs. Added Titanium F100 and F256 packages to interface floorplans. Added F100 and F256 packages for Ti35 and Ti60 FPGAs in **Table 1: Titanium Interface Block Support by Package** on page 9. Added HyperRAM for Ti35 and Ti60 in F100S3F2 packages to **Table 1: Titanium Interface Block Support by Package** on page 9.
February 2024	4.2	Removed 'ns', added note and description in table Timing Tab of Using the MIPI D-PHY TX section. (DOC-1699 and DOC-1700) Updated description for HSIO block DLY_INC signal. (DOC-1697)

Date	Version	Description
January 2024	4.1	Added Ti135 and Ti240 to **Interface Blocks** on page 9 and **Package/ Interface Support Matrix** on page 10. (DOC-1661) Corrected OUTCLK connection in **Figure 23: I/O Interface Block** on page 68. (DOC-1630) Added 1.35 V HSIO support for Ti135, Ti240, and Ti375.
December 2023	4.0	Added fractional PLL content. Combined all PLL topics into a single chapter. Added hardened RISC-V block content. Updated Design Check topics for new/updated messages. Added topic on clocking interface blocks. (DOC-1412) Removed the figures for the emulated MIPI groups by package. Instead refer to the Titanium Packaging User Guide.
October 2023	3.2	Updated Create a MIPI TX Interface topic by adding reference clock and feedback mode options. (DOC-1427) Added Drive Strength setting and design checks for HyperRAM block. (DOC-1444) Added DDR interface Pin Swizzling options and updated DDR design checks. (DOC-1445) Updated GPIO and LVDS block design checks. (DOC-1481)
August 2023	3.1	Updated support for G400 packages. (DOC-1394)
June 2023	3.0	Improved MIPI RX function description, MIPI RX interface block diagram, and MIPI RX lane block diagrams. (DOC-1173) Added slvs option for HSIO configured as LVDS blocks. (DOC-1190) Updated PLL's Invert Output Clock to Output Clock Inversion option which allows the inversion of output clock individually. (DOC-941) Updated SPI Flash Interface Designer settings. (DOC-1296) Updated MIPI D-PHY TX Interface Designer settings. (DOC-1178) Added PLL SSC block. (DOC-1178) Updated design checks messages for Clock Control, DDR Errors, MIPI DPHY Errors, and PLL Errors.
June 2023	2.9	Updated DDR_DM signal description and added Enable DBI options. (DOC-1322)
April 2023	2.8	Added LVDS RX DBG signals. (DOC-1124) Updated PLL LOCKED signal description. (DOC-1208) Added note to state that the PLL tracks the reference clock input frequency accuracy. (DOC-1179) Added note about using LVDS blocks from the same side of the FPGA to minimize skew. (DOC-1150) Updated DDR DRAM interface input clock to include description for J361, J484, and G529 packages. (DOC-1209) Added PLL Interface Designer Settings and updated description for CLKOUT when driving core logic for Ti35 and Ti60 FPGAs. (DOC-1130) Added DPA to LVDS RX options table. (DOC-922) Removed DDR block AXI width option. (DOC-1210) Updated PLL RSTN signal description about de-asserting only when CLKIN is stable. (DOC-1226)
February 2023	2.7	Corrected PLL_SSC_EN MIPI TX D-PHY signal notes. (DOC-1101)
December 2022	2.6	Updated support for J361, J484, and G529 packages. Updated MIPI DPHY TX and DDR Interface Designer Settings.

Date	Version	Description
October 2022	2.5	Updated DDR DRAM interface signals. Updated DDR DRAM Interface Designer Settings.
September 2022	2.4	Updated PLL clock for DDR DRAM block. (DOC-881) Corrected MIPI RX Lane Block Diagram. (DOC-878) Removed GCTRL and RCTRL. (DOC-895) Added topics on Package Planner. Corrected AWID_x, AWREADY_x, ARADDR_x, and AWADDR_x DDR signals directions and widths. (DOC-907) Removed PLL_EXTFB from alternative input. (DOC-849)
July 2022	2.3	Corrected floorplan diagrams.
July 2022	2.2	Corrected the LVDS maximum speed. (DOC-807) Removed reference to T13 and T20. (DOC-807) Updated MIPI D-PHY port names. (DOC-782) Added I/O banks by package information for Ti90, Ti120, and Ti180. (DOC-821) Updated DDR pad names. Added M361, L484, M484, and F529 floorplans.
April 2022	2.1	Corrected RD and RST signal directions in MIPI RX Lane Block Diagram. Corrected description for differential TX static programmable delay. (DOC-786) Updated HyperRAM clock rate and double data rate specs. (DOC-793) Corrected missing link and added pointer for list of clock sources in Global Buffer Configuration table.
February 2022	2.0	Added Titanium DDR block interface description. Added Titanium MIPI DPHY interface block description. HVIO I/O banks support dynamic voltage shifting. (DOC-444) Added MIPI RX clock groups for F484 package. Added interface floorplan for F484 package. New design rules: io_bank_rule_mode_sel, io_bank_rule_dyn_voltage. Updated label for Ti60 W64 pin A7. (DOC-651)
November 2021	1.1	Updated PLL Block Diagram to indicate F_{PLL}. Updated JTAG mode connection diagram. (DOC-546) Updated PLL phase-shift descriptions. (DOC-570) PLL outputs lock on the negative clock edge. (DOC-552) Added example PLL zero-delay buffer implementation. (DOC-551) Added an example for the PLL outputs for the Create a MIPI TX Interface topic. (DOC-580) New design rule: clock_rule_lvds_rx_clock_source. This rule is effective with Efinity patch v2021.1.4.10. (DOC-589) New design rules: clock_rule_pll_ref_clock_lvds_rx, pll_rule_pll_freq, lvds_rule_tx_clock_region, lvds_rule_rx_clock_region, lvds_rule_rx_dpa_serial, and mipi_ln_rule_tx_clock_region.
June 2021	1.0	Initial release for Efinity software v2021.1.