



PLL Dynamic Reconfiguration Core User Guide

UG-CORE-PLL-DYN-RECFG-v1.4
April 2026
www.efinixinc.com



Contents

Introduction.....	3
Features.....	3
Device Support.....	3
Resource Utilization and Performance.....	4
Release Notes.....	4
Functional Description.....	5
PLL Dynamic Reconfiguration Mode.....	5
Reference Clock Selection.....	6
Clock Sources.....	6
Multi-Configuration Mode Handshaking.....	7
Single Configuration Mode Handshaking.....	8
Revert to PCR Settings Handshaking.....	8
Ports.....	9
Efinity Compilation.....	11
Interface Designer Settings.....	12
Set Up the PLL and Enable the Dynamic Reconfiguration.....	12
Generate the RAM Hex File.....	12
IP Manager.....	14
Top Level Integration.....	16
Special Handling for Efinity Attribute.....	16
Customizing the PLL Dynamic Reconfiguration Core.....	17
PLL Dynamic Reconfiguration Example Design.....	18
Virtual I/O Debugger Settings.....	20
PLL Dynamic Reconfiguration Testbench.....	20
Debugging: PLL Locking.....	21
Revision History.....	22

Introduction

The PLL Dynamic Reconfiguration core allows users to change the PLL settings on an Efinix FPGA, without changing the settings in the Interface Designer, re-compiling, and re-generating the bitstream file. This IP core provides an efficient way to change the PLL settings on the fly after initial boot-up, without having to re-configure the FPGA.

Use the IP Manager to select IP, customize it, and generate files. The PLL Dynamic Reconfiguration core has an interactive wizard to help you set parameters. The wizard also has options to create a testbench and/or example design targeting an Efinix[®] development board.

Features

The PLL Dynamic Reconfiguration core includes the following features:

- Allows up to 85 PLL settings to be stored and reconfigure the PLL
- Allows the FPGA to revert to the original PLL settings in the bitstream file⁽¹⁾
- Single configuration and multiple configuration mode

Device Support

PLL Dynamic Reconfiguration core is supported in Titanium and Topaz FPGA with Fractional PLL. Refer to the latest devices in the Interfaces User Guides in the [Support Center](#) for the availability of the Fractional PLL core and PLL Dynamic Reconfiguration core in each Efinix device.

⁽¹⁾ The original PLL settings are also known as Peripheral Configuration Register (PCR) settings.

Resource Utilization and Performance



Note: The resources and performance values provided are based on some of the supported FPGAs. These values are just guidance and may change depending on the device resource utilization, design congestion, and user design.

Table 1: Titanium Resource Utilization

FPGA	Mode	Logic Elements (Logic, Adders, Flipflops, etc.)	Memory Block	DSP Block	Efinity® Version ⁽²⁾
Ti375 C529 C4	Multiple Configuration	455/370,137 (0.12%)	2/2,688 (0.07%)	0/1,344 (0%)	2024.2
	Single Configuration	329/370,137 (0.09%)	0/2,688 (0%)	0/1,344 (0%)	

Release Notes

You can refer to the IP Core Release Notes for more information about the IP core changes. The IP Core Release Notes are available on the [Efinity Downloads](#) page under each Efinity software release version.



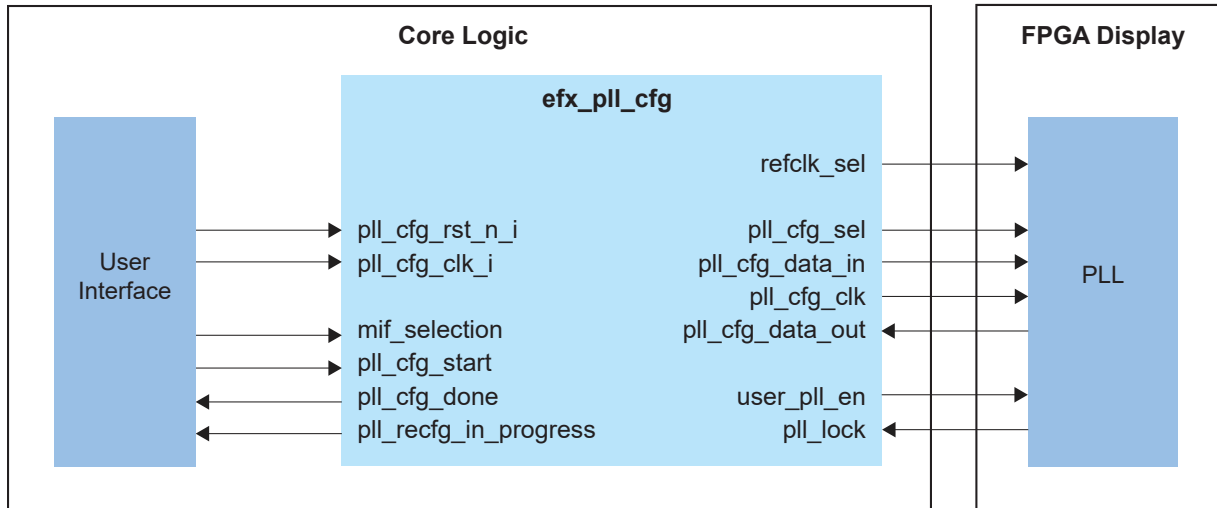
Note: You must be logged in to the Support Center to view the IP Core Release Notes.

⁽²⁾ Using System Verilog 2005.

Functional Description

The PLL Dynamic Reconfiguration core is positioned between the user's logic and the PLL to facilitate the dynamic reconfiguration scheme.

Figure 1: PLL Dynamic Reconfiguration Block Diagram



PLL Dynamic Reconfiguration Mode

The PLL Dynamic Reconfiguration core uses stored configuration settings during operation. Therefore, before beginning the operation, you need to store the intended configuration settings. The PLL Dynamic Reconfiguration core has built-in RAM to store up to 85 PLL configuration settings.

- If you have multiple configuration settings, select **Multiple Configuration** when configuring the core.
- If you have only one configuration setting, select **Single Configuration**. In **Single Configuration** mode, when you generate the IP core, it is optimized to reduce the RAM to registers.

To customize the configuration settings, you need to use the Efinity Interface Designer. Refer to [Interface Designer Settings](#) on page 12.

The PLL Dynamic Reconfiguration core has the capability to allow you to revert to the settings in the original bitstream. The PCR settings are configured during the FPGA power-up and they are the original PLL settings that you made in the Interface Designer.

For more information, refer to [Multi-Configuration Mode Handshaking](#) on page 7, [Single Configuration Mode Handshaking](#) on page 8, and [Revert to PCR Settings Handshaking](#) on page 8.

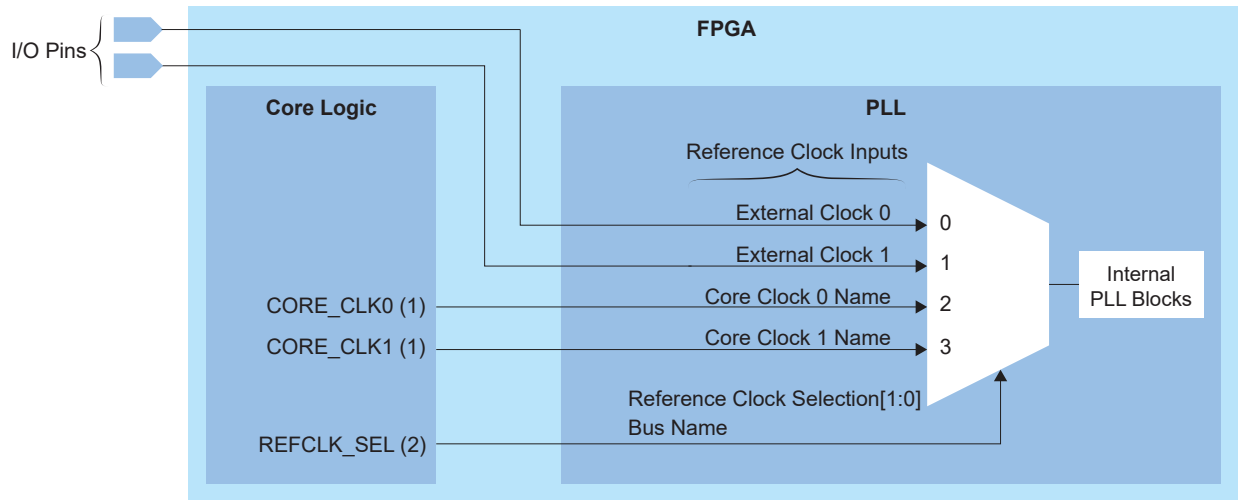
Reference Clock Selection

You can change the PLL's reference clock by controlling the reference clock selection port, REFCLK_SEL. When the PLL Dynamic Reconfiguration is enabled, the core drives the REFCLK_SEL port as illustrated in **Figure 2: PLL Reference Clock Selection** on page 6.



Note: The REFCLK_SEL port is only active after the FPGA powers up, i.e., after the FPGA enters user mode. Hence, it is essential that the initial value of this REFCLK_SEL port matches the bitstream's original PCR settings to prevent unintentional reference clock input switching.

Figure 2: PLL Reference Clock Selection



Notes:

1. CORE_CLK1 and CORE_CLK0 are user-defined names and connect to the core logic by name.
2. REFCLK_SEL is user defined and is driven by the PLL Dynamic Reconfiguration core.

Clock Sources

The core has two clock domains:

- Fast clock, pll_cfg_clk_i
- Slow clock, pll_cfg_clk

Table 2: PLL Dynamic Reconfiguration Core Clock Sources

Clock	Direction	Frequency (MHz)	Description
pll_cfg_clk_i	Input	50 - 300	Fast clock. This is a free running clock.
pll_cfg_clk	Output (Generated Clock)	25 - 150	Slow clock. This clock is generated from IP and must connect to PLL.



Important: pll_cfg_clk is generated from pll_cfg_clk_i. Hence, you need to add create_generated_clock constraint into the SDC. Refer to the SDC file in the example design. The port and pin names are dependent on your design.

Multi-Configuration Mode Handshaking

To start the PLL Dynamic Reconfiguration operation, assert the `p11_cfg_start` port as illustrated in **Figure 3: Multi-Configuration Mode Handshaking Waveform** on page 7. This assertion allows the PLL Dynamic Reconfiguration core to deassert `user_pll_en` to reset the PLL, resulting in the PLL losing lock. The `p11_recfg_in_progress` port asserts at the same time to indicate that the PLL Dynamic Reconfiguration operation has started.

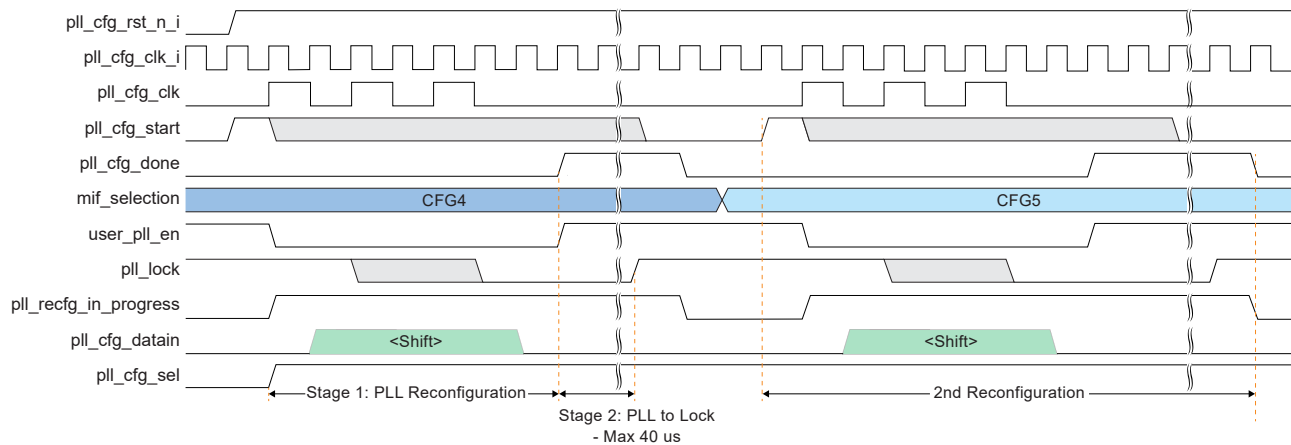
Note: The `p11_cfg_start` port should be pulse-based and assert for 1 clock cycle to begin reconfiguration. If this port is high beyond stage 2 (as shown in **Figure 3: Multi-Configuration Mode Handshaking Waveform** on page 7), it is interpreted as an assertion to begin the second reconfiguration.

Depending on the `mif_selection` value, the PLL Dynamic Reconfiguration core starts to fetch the intended configuration settings from the RAM. It shifts the settings into the PLL. When the configuration finishes the shifting, the `p11_cfg_done` port asserts, indicating the end of the first reconfiguration stage. The second reconfiguration stage follows automatically; the PLL Dynamic Reconfiguration core re-asserts `user_pll_en` to re-enable the PLL with the newly loaded settings. The PLL is expected to regain lock state within 40 μ s. Upon assertion of `p11_lock`, both `p11_recfg_in_progress` and `p11_cfg_done` are deasserted, indicating the end of the second stage and the end of the PLL Dynamic Reconfiguration core's operation.

You can repeat the first stage and second stage to perform another round of reconfiguration if needed.

Note: The `p11_cfg_done` and `p11_recfg_in_progress` ports are useful indicators of the dynamic reconfiguration status.

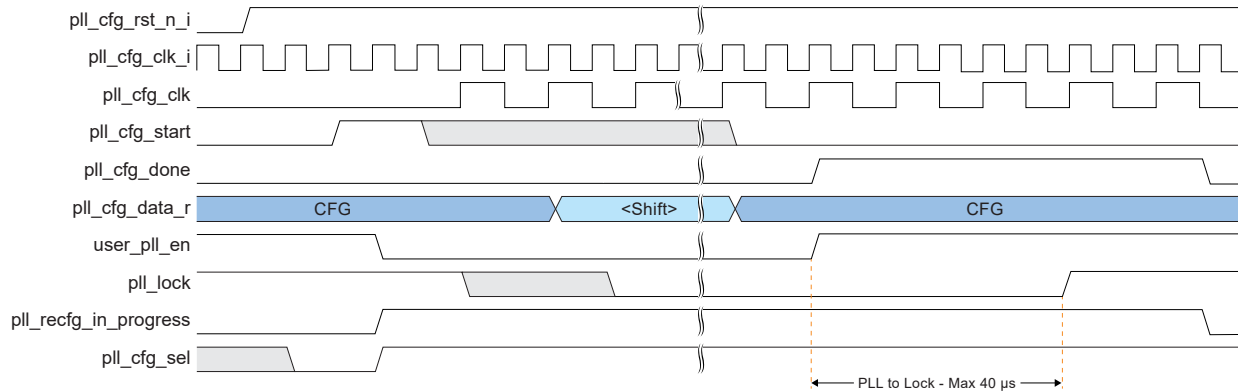
Figure 3: Multi-Configuration Mode Handshaking Waveform



Single Configuration Mode Handshaking

Single configuration mode is similar to multiple configuration mode. Unlike multi-configuration mode, single configuration mode does not use the `mif_selection` port. Set this port to 0. Refer to **Figure 4: Single Configuration Mode Handshaking Waveform** on page 8, which illustrates the handshaking in single configuration mode.

Figure 4: Single Configuration Mode Handshaking Waveform

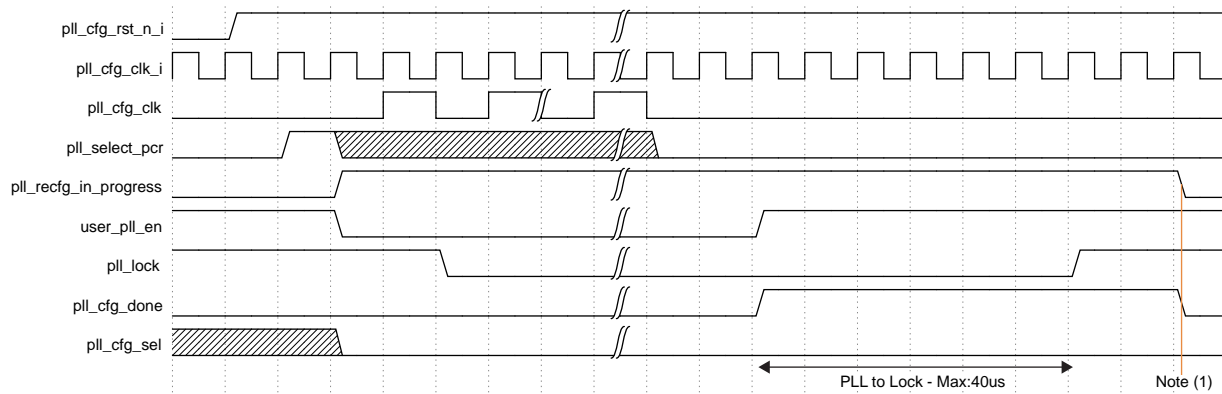


Revert to PCR Settings Handshaking

To revert to PCR setting, assert the `pll_select_pcr`. The PCR settings are the settings configured during the FPGA power-up. The assertion of `pll_select_pcr` results in the deassertion of `user_pll_en` and the reset of the PLL. Similarly, the `pll_recfg_in_progress` asserts to indicate that the reverting to the PCR setting has begun.

The assertion of `pll_cfg_done` indicates that the PLL is completely unlocked, the PCR settings are effective, and the PLL calibration to relock has begun.

Figure 5: Handshake for PLL Revert Feature



Notes: (1) The assertion of `pll_lock` together with the de-assertion of `pll_recfg_in_progress` indicate that PLL has re-locked with the PCR Settings.

Ports

Table 3: PLL Dynamic Reconfiguration Core Controller Ports

Port	Direction	Width	Clock Domain	Description
pll_cfg_rst_n_i	Input	1	RESET	Reset the PLL dynamic reconfiguration. Should only assert (if any) when pll_recfg_in_progress is low. Any assertion of reset/interruption during the PLL dynamic reconfiguration, i.e., when pll_recfg_in_progress is set to 1, may jeopardize the final configuration and cause the PLL to be non-functional or unable to lock. The pll_cfg_rst_n_i ports must be initialized to 0. They can only be de-asserted at least one clock (pll_cfg_clk_i) cycle later.
pll_cfg_clk_i	Input	1	CLOCK	Clock source to the PLL Dynamic Reconfiguration core. This is a free running clock.
pll_select_pcr	Input	1	pll_cfg_clk_i	Enable configuration to switch from the dynamic reconfiguration settings to the original bitstream PCR settings. Refer to Figure 5: Handshake for PLL Revert Feature on page 8.
mif_selection	Input	7	pll_cfg_clk_i	RAM Hex File configuration pattern ID. . Applicable only in multi-configuration mode. Refer to Figure 3: Multi-Configuration Mode Handshaking Waveform on page 7.
pll_cfg_start	Input	1	pll_cfg_clk_i	Assert this signal to start PLL dynamic reconfiguration. The pll_cfg_start must be initialized to 0 for at least 1 clock cycle of pll_cfg_clk_i. Refer to Figure 3: Multi-Configuration Mode Handshaking Waveform on page 7 and Figure 4: Single Configuration Mode Handshaking Waveform on page 8.
pll_cfg_done	Output	1	pll_cfg_clk_i	Indicates of the completion of PLL dynamic reconfiguration. When the signal is asserted, you cannot revert to the original bitstream PCR settings (pll_select_pcr). Refer to Figure 5: Handshake for PLL Revert Feature on page 8. After the assertion of pll_cfg_done, the pll locks within 40 μ s and pll_recfg_in_progress is deasserted after the PLL is locked. If the PLL is not locked after 40 μ s, refer to the Debugging: PLL Locking on page 21 section.

Port	Direction	Width	Clock Domain	Description
pll_recfg_in_progress	Output	1	pll_cfg_clk_i	Indicates that PLL dynamic reconfiguration is in progress. Refer to Figure 3: Multi-Configuration Mode Handshaking Waveform on page 7, Figure 4: Single Configuration Mode Handshaking Waveform on page 8, and Figure 5: Handshake for PLL Revert Feature on page 8.
pll_cfg_data_o	Output	1	pll_cfg_clk	Serial data output from the dynamic reconfiguration registers in the PLL. For debug/monitoring purposes.

Table 4: PLL Dynamic Reconfiguration Core Controller Ports for Interfacing with the Interface Designer

Port	Direction	Width	Clock Domain	Description
pll_cfg_clk	Output	1	CLOCK	Active/toggles only to clock the shift of pll_cfg_data_in. Disable when not in operation. This port should be connected to the PLL in the Interface Designer.
pll_cfg_data_out	Input	1	pll_cfg_clk	Serial data output from the dynamic reconfiguration registers in the PLL. This port should be connected to the PLL in the Interface Designer.
pll_cfg_data_in	Output	1	pll_cfg_clk	Serial data input to configure the PLL. This port should be connected to the PLL in the Interface Designer.
pll_cfg_sel	Output	1	pll_cfg_clk_i	Select signal to select the configuration between PCR and soft IP dynamic reconfiguration. This port should be connected to the PLL in the Interface Designer.
refclk_sel	Output	2	pll_cfg_clk_i	Configuration for the reference clock of the PLL. This port should be connected to the PLL in the Interface Designer.
pll_lock	Input	1	Async	Pseudo static. Indicator of PLL lock. This port should be connected to the PLL's pll_lock port in the PLL Clock Calculator wizard in the Interface Designer.
user_pll_en	Output	1	pll_cfg_clk_i	Set user_pll_en = 0 to disable the PLL before any reconfiguration. Set user_pll_en = 1 after the dynamic configuration completes to re-enable and restart the PLL locking. This port should be connected to the PLL reset in the PLL Clock Calculator in the Interface Designer.

Efinity Compilation

The PLL Dynamic Reconfiguration core is positioned between the user's logic and the PLL to facilitate the PLL's dynamic reconfiguration as shown in **Figure 1: PLL Dynamic Reconfiguration Block Diagram** on page 5. The flow for setting up the PLL Dynamic Reconfiguration core, the PLL, and your user logic requires the following steps:

1. Setup the PLL and enable **Dynamic Reconfiguration** using the Interface Designer.
2. Configure and generate the PLL Dynamic Reconfiguration core using the IP Manager.
3. Create a top-level module to integrate the PLL (from step 1), the PLL Dynamic Reconfiguration core (from step 2), and your user logic.



Note: When using the Interface Designer, note that:

- Signal names are case-sensitive.
- All user-defined ports in the Interface Designer connect by name to the core logic. Therefore, all port names in the Interface Designer must match (in name and case) the signal names in your design to connect effectively.

Interface Designer Settings

Before generating the PLL Dynamic Reconfiguration core in the Efinity IP Manager, you must set up a PLL Block and generate a RAM Hex File in the Interface Designer. The RAM Hex File contains one or more PLL configuration settings to be stored in the built-in RAM in the PLL Dynamic Reconfiguration core.

Set Up the PLL and Enable the Dynamic Reconfiguration

For more information on setting up the PLL, refer to PLL interface in the Interfaces User Guide. To enable the PLL Dynamic Reconfiguration core, follow these steps:

1. In the Interface Designer, add a PLL block.
2. Click **Automated Clock Calculation** to open the PLL Clock Calculator.
3. Specify a reference clock frequency.
4. Turn on the **Reset Pin** option and enter the name `user_pll_en`.
5. Turn on the **Locked Pin** option and enter the name `pll_lock`.
6. Click **Finish** to close the PLL Clock Calculator.
7. Click the **Advanced Settings** tab.
8. Turn on **Enable** under **Dynamic Reconfiguration**.
9. Enter the reference clock names.
10. Create GPIO blocks for any external clocks and core clocks as needed. All clock names should be the same as the ones that you are using in your design. Names are case sensitive.

Generate the RAM Hex File

To generate a RAM Hex File, follow these steps in the Interface Designer for the PLL block you just created:

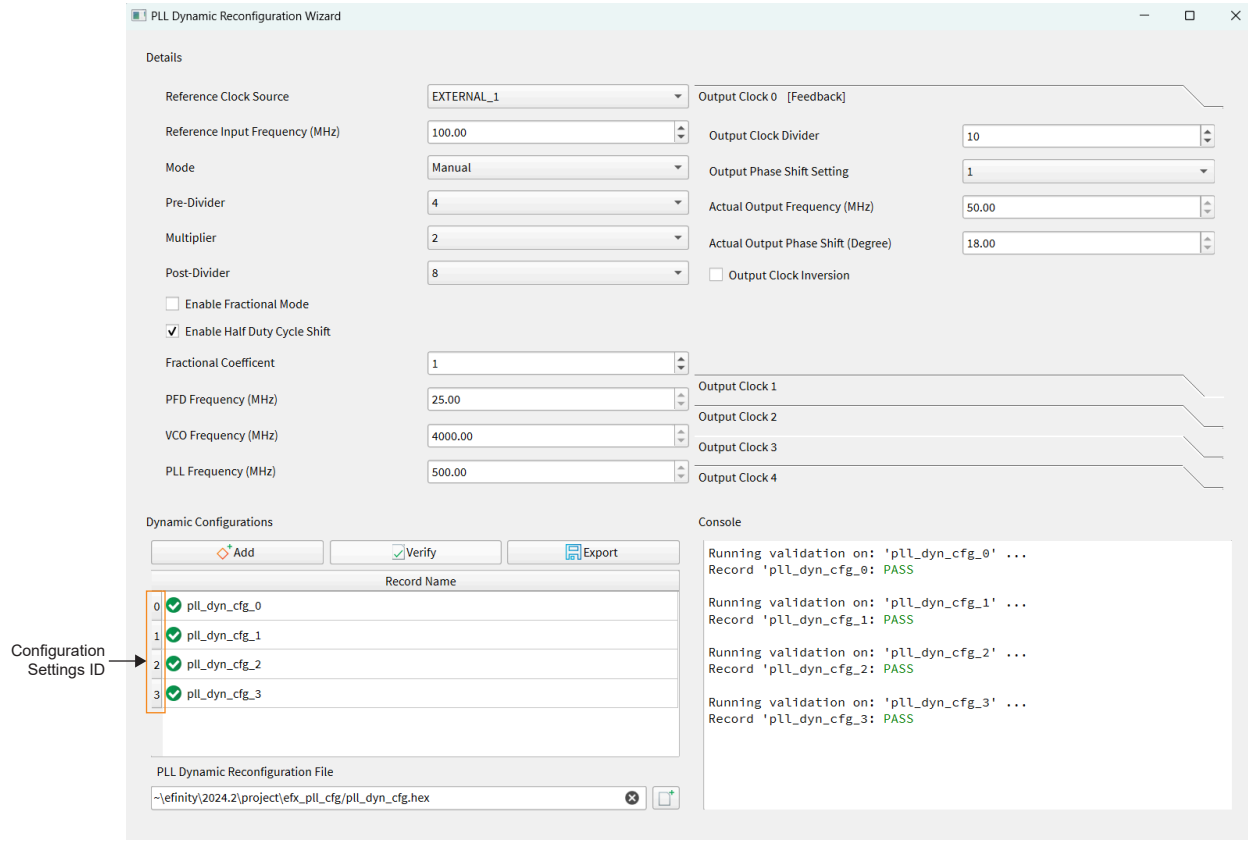
1. Click **Advanced Settings > Reconfiguration Wizard** to open the PLL Dynamic Reconfiguration Wizard.
2. Click **Add**. In the **Record Name** box, a new configuration setting names would appear, along with the numbering on the left column.



Note: The numbering on the left column is the ID of the configuration settings. During the **Dynamic Reconfiguration with Multiple Configuration**, you need to drive the signal `mif_selection` based on this ID.

3. Click on the configuration file that you have just added and start customizing the configuration settings.
4. Once you finish customizing the configuration settings, click **Verify**.
5. Repeat steps 2, 3, and 4 to add as many configurations (up to 85 configurations) as desired.
6. Enter a filename in the **PLL Dynamic Reconfiguration File** box or use the default filename.
7. Finally, click **Export**.

Figure 6: PLL Dynamic Reconfiguration Wizard



IP Manager

The Efinity® IP Manager is an interactive wizard that helps you customize and generate Efinix® IP cores. The IP Manager performs validation checks on the parameters you set to ensure that your selections are valid. When you generate the IP core, you can optionally generate an example design targeting an Efinix development board and/or a testbench. This wizard is helpful when you use several IP cores, multiple instances of an IP core with different parameters, or the same IP core across different projects.



Note: Not all Efinix IP cores include an example design or a testbench.

Generating the PLL Dynamic Reconfiguration Core with the IP Manager

The following steps explain how to customize an IP core with the IP Configuration wizard.

1. Open the IP Catalog.
2. Choose **Foundation IP > PLL Dynamic Reconfiguration** core and click **Next**. The **IP Configuration** wizard opens.
3. Enter the module name in the **Module Name** box.



Note: You cannot generate the core without a module name.

4. Customize the IP core using the options shown in the wizard. For detailed information on the options, refer to the [Customizing the PLL Dynamic Reconfiguration Core](#) on page 17 section.
 5. In the **Initial Reference Clock Setting**, key in the reference clock setting.
-
- Note:** This reference clock setting must match the clock source of the PCR setting to prevent unintended clock switching during the FPGA power up. Refer to [Reference Clock Selection](#) on page 6.
6. (Optional) In the **Deliverables** tab, specify whether to generate an IP core example design targeting an Efinix® development board and/or testbench. These options are turned on by default.
 7. (Optional) In the **Summary** tab, review your selections.
 8. Click **Generate** to generate the IP core and other selected deliverables.
 9. In the **Review configuration generation** dialog box, click **Generate**. The Console in the **Summary** tab shows the generation status.



Note:

- The PLL instance name is the same name you have created for the targeted PLL in Interface Designer.
- Ensure that the **Initial Reference Clock Setting** is the same with what you have selected in the interface designer.
- You can disable the **Review configuration generation** dialog box by turning off the **Show Confirmation Box** option in the wizard.

10. When generation finishes, the wizard displays the **Generation Success** dialog box. Click **OK** to close the wizard.

The wizard adds the IP to your project and displays it under **IP** in the Project pane.

Generated Files

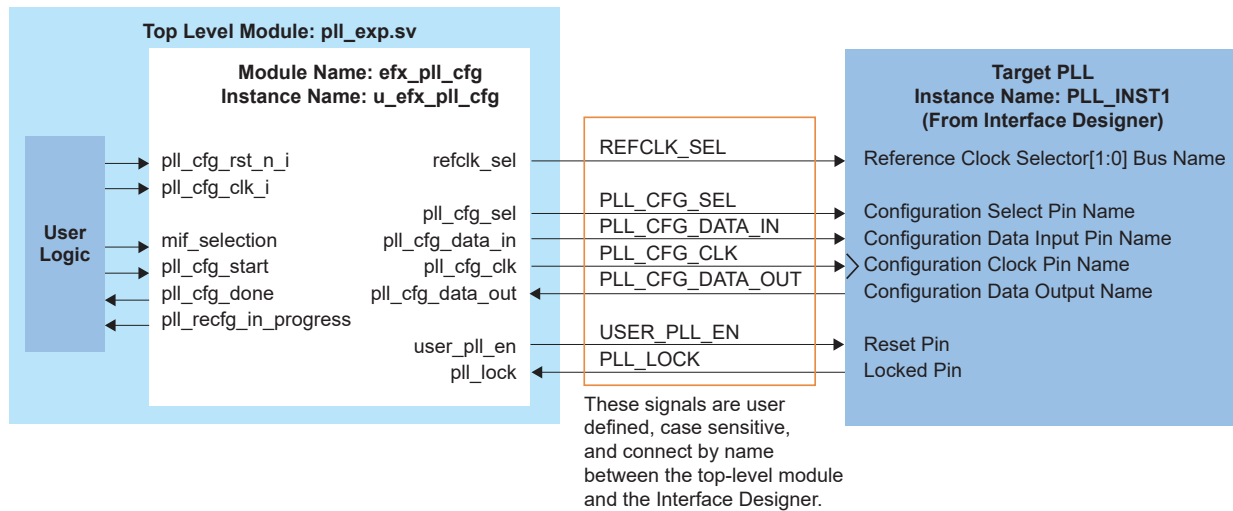
The IP Manager generates these files and directories:

- **<module name>_define.svh**—Contains the customized parameters.
- **<module name>_tpl.sv**—Verilog HDL instantiation template.
- **<module name>_tpl.vhd**—VHDL instantiation template.
- **<module name>.sv**—IP source code.
- **settings.json**—Configuration file.
- **<kit name>_devkit**—Has generated RTL, example design, and Efinity® project targeting a specific development board.
- **Testbench**—Contains generated RTL and testbench files.

Top Level Integration

After setting up the PLL using the Interface Designer and generating the PLL Dynamic Reconfiguration core, you must create a top-level module to integrate both the PLL and the PLL Dynamic Reconfiguration core. Refer to [Figure 7: Top-Level Module for Efinity Compilation](#) on page 16.

Figure 7: Top-Level Module for Efinity Compilation



Special Handling for Efinity Attribute

For Efinity to synthesize the core-generated clock connection to the PLL correctly, you need to use the `syn_preserve` synthesis attribute to keep the core-generated `pll_cfg_clk` signal during optimization.

You need to remove the original output port declaration of `PLL_CFG_CLK`. At the same time, you need to include the `syn_preserve` synthesis attribute as indicated by the bold fonts in the following [Figure 8: Handling Efinity Attribute](#) on page 16.

Figure 8: Handling Efinity Attribute

```

module pll_exp (
  ...
  output logic PLL_CFG_CLK
);
(* syn_preserve = "true" *) logic PLL_CFG_CLK; //attribute to be added
  ...
endmodule

```



Learn more: Refer to the example design, `<path to project>\ip\<module>\Ti375C529_devkit\pll_exp.sv`

Customizing the PLL Dynamic Reconfiguration Core

The core has parameters so you can customize its function. You set the parameters in the **General** tab of the core's IP Configuration window.

Table 5: PLL Dynamic Reconfiguration Core Parameter

Parameter	Options	Description
Configuration Mode	Multi Configuration, Single Configuration	Multi Configuration: Select multi-configuration if you want to use more than one PLL configuration setting. Single Configuration: Select single configuration to optimize the resources if you are only using one configuration.
RAM Hex File Path	-	For multiple configuration, browse to the RAM Hex file path. Refer to Generate the RAM Hex File on page 12 topic to create a RAM Hex file if it is unavailable. To generate the IP successfully, make sure you select the correct file.
Configuration Setting	-	For single configuration, you need to enter the configuration setting (which is a string) into this box. To obtain the configuration setting string, open the <generated RAM Hex file>.dyn_cfg.rpt file. This report file is generated when the <generated RAM Hex file>.hex is generated.
PLL Instance Name	-	Enter the PLL block name you created in the Interface Designer for the PLL that you want to reconfigure.
Reference Clock Setting	External Clock 0, External Clock 1, Core Clock 0, Core Clock 1	Initial reference clock. Default: External clock 0 This option is the same as the Interface Designer's reference clock settings.

PLL Dynamic Reconfiguration Example Design

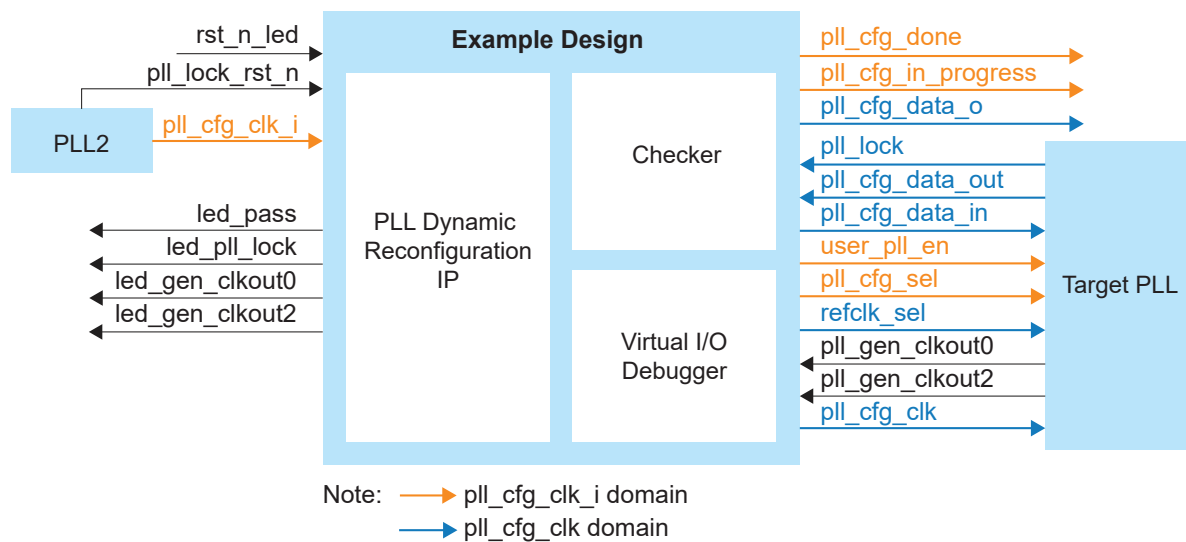
You can choose to generate the example design when generating the core in the IP Manager Configuration window. Compile the example design project and download the **.hex** or **.bit** file to your board. To generate example design, the **Example Design Deliverables Option** signal must be enabled.



Important: Efinix tested the example design generated with the default parameter options only.

Efinix provides the following example design.

Figure 9: PLL Dynamic Reconfiguration Example Design



The example design targets the Titanium Ti375 C529 Development Board.

In the example design directory, you can use the RAM Hex File file as provided in the folder. For multiple configuration, ensure that the file path directory is correct for your design. This example design also has a plug-in interface known as **Virtual I/O Debugger**. This interface allows you to download the bitstream to your device. After downloads are complete, you can use the **Virtual I/O Debugger** interface to control some of the control signals of the example design. To do the settings, refer to **Virtual I/O Debugger Settings** on page 20.

To implement the example design:

1. Create a new project.
2. Generate the IP.
3. Open the project file (.xml) in `<path to project>\ip\<ip-module-name>\Ti375C529_devkit\pll_exp.xml`.



Note: In `<path to project>\ip\<ip-module-name>\Ti375C529_devkit\<ip-module-name>.sv`, ensure that the `<RAM Hex File name>.hex` is directed correctly at the `RAM_INIT_FILE` parameter.

4. Compile the design.
5. Download the design to your Titanium Ti375 C529 Development Board.

Table 6: Example Design Input and Output

Input / Output	Resource	Dev Kit Coordinate	Description
led_pass	GPIOT_N_64	LED13	This LED indicates whether the configuration setting is shifted to the targeted PLL correctly.
led_pll_lock	GPIOT_N_19	LED12	This LED indicates the PLL lock status of the targeted PLL.
led_gen_clkout0	GPIOT_N_50	LED11	This LED blinks based on the frequency of counter_clkout0.
led_gen_clkout2	GPIOL_24	LED10	This LED blinks based on the frequency of counter_clkout2.
rst_n_led	GPIOL_52	SW3	Reset the counter of led_gen_clkout0 and led_gen_clkout2.



Important: In multiple configuration mode, refer to file `/<IP name>/Ti375C529_devkit/<IP name>.sv`. You must ensure that the RTL's `RAM_INIT_FILE` parameter is pointed to the correct path for your RAM Hex File file before compiling.

Table 7: Example Design Project Files

File Name	Description
pll_exp.sv	Example design of top-level wrapper.
<user_given_ip_name>.sv	The generated encrypted PLL Dynamic Reconfiguration file based on user configuration in Efinity IP Manager.
cfg_checker.sv	Module for checking configuration shifting.
debug_top.v	Verilog file for EFX debug module.
constraint.sdc	Constraint file for example design.
pll_dyn_cfg.hex	Example of a generated RAM Hex File.
debug_profile.json	Virtual I/O debugger core file. Load this file in the Efinity Virtual I/O debugger to customize the example design. See Virtual I/O Debugger Settings on page 20.



Note: In Efinity 2025.1 and later, the PLL Dynamic Reconfiguration core example design has added a new synthesis project setting to include the directory path relative to the user's project directory where the PLL Dynamic Reconfiguration core is generated.

Virtual I/O Debugger Settings

The example design includes Efinity Virtual I/O Debugger core for customizing and monitoring the design. The following table describes the Virtual I/O sources and descriptions.

Table 8: Efinity Virtual I/O Debugger Settings

Name	Width	Radix	Description
refclk_sel	2	Hex	This is a probe to show the current reference clock.
pll_lock	1	Bin	This probe indicates the targeted PLL lock status.
user_pll_en	1	Bin	This probe indicates the PLL enable signal that drives the PLL.
pll_cfg_start	1	Bin	A user input signal to start the PLL dynamic reconfiguration. Efinix recommends that you control this signal by selecting Active-High Button from a drop-down list in the control column.
mif_selection	7	Hex	An input signal to select the PLL configuration.
pll_select_pcr	1	Bin	A user input signal to revert the PLL dynamic reconfiguration to the PCR setting. Efinix recommends that you control this signal by selecting Active-High Button from a drop-down list in the control column.

PLL Dynamic Reconfiguration Testbench

You can choose to generate the testbench when generating the core in the IP Manager Configuration window. To generate testbench, the **Optional Signals** option must be enabled.



Note: You must include all **.sv** or **.v** files generated in the **/testbench** directory in your simulation.



Important: Efinix tested the testbench generated with the default parameter options only.



Important: In multiple configuration mode, go to **/Testbench/modelsim/<ip_name>.sv** to ensure that the RAM Hex File file path is correct before running the simulation.

Efinix provides a simulation script for you to run the testbench quickly using the Modelsim software. To run the Modelsim testbench script, run `vsim -do modelsim.do` in a terminal application. You must have Modelsim installed on your computer to use this script.

Debugging: PLL Locking

The estimated time for the PLL to lock is 40 μ s. You can refer to the *PLL to lock* stage in [Figure 3: Multi-Configuration Mode Handshaking Waveform](#) on page 7, [Figure 4: Single Configuration Mode Handshaking Waveform](#) on page 8, and [Figure 5: Handshake for PLL Revert Feature](#) on page 8. If the *PLL to lock* stage prolongs beyond 40 μ s, the PLL is considered not locking with the post-dynamic reconfiguration settings. The following are the possible reasons, but not limited to:

- The RAM Hex File content is compromised.
- The reference clock source is compromised.
- Non-compliant handshake, especially signals `p11_cfg_start` and `mif_selection`.
- Unexpected deassertion of `p11_cfg_rst_n_i`, especially when the PLL reconfiguration process is in progress (as indicated by `p11_recfg_in_progress`).



Note:

1. The ability of the PLL to lock relies on empirical configuration settings. If the PLL fails to lock, you must check whether the content of the RAM Hex file has been manually edited. You can generate a valid RAM Hex file from the Interface Designer and cross-check against any failing RAM Hex file configurations.
 2. The *PLL to lock* stage starts at the assertion of signal `p11_cfg_done` and ends at the assertion of `p11_lock`.
 3. If the PLL lock time (denoted by *PLL to lock* stage) exceeds 40 μ s, you may assert `p11_cfg_rst_n_i` to reset the internal state machine. After the reset, you may either restart the configuration process or begin debugging using the methods described.
-

Revision History

Table 9: Revision History

Date	Document Version	IP Version	Description
April 2026	1.4	1.2	Corrected the mif_selection width to 7 from 1 in Table 8: Efinity Virtual I/O Debugger Settings on page 20. (DOC-3002)
September 2025	1.3	1.1	Added back Topaz in Device Support. (DOC-2686)
August 2025	1.2	1.1	Corrected Figure PLL Reference Clock Selection. (DOC-2660) Updated Debugging: PLL Locking topic.
May 2025	1.1	1.1	Removed Topaz and added note in Device Support. (DOC-2436) Changed title of Special Handling for Efinity Symbols to Special Handling for Efinity Attribute. Updated figure Handshake for PLL Revert Feature and Top-Level Module for Efinity Compilation. Updated Special Handling for Efinity Attribute topic. Added column Reference and Dev Kit Coordination in table Example Design Input and Output. Updated Multi-Cofiguration Mode Handshaking, Revert to PCR Settings Handshaking, and Special Handling for Efinity® Attribute topic. Added note for table Example Design Project Files and Clock Sources.
December 2024	1.0	1.0	Initial release. Added IP Version in Revision History. (DOC-2185)