Processor System Reset Module v5.0

LogiCORE IP Product Guide

PG164 November 18, 2015

XILINX®

Table of Contents

IP Facts

Chapter 1: Overview

Chapter 2: Product Specification

Chapter 3: Designing with the Core

Chapter 4: Customizing and Generating the Core

Chapter 5: Constraining the Core

Chapter 6: Simulation

Chapter 7: Synthesis and Implementation

Chapter 8: Example Design

Chapter 9: Test Bench

Appendix A: Migrating and Upgrading

Appendix B: Debugging

Appendix C: Additional Resources

Introduction

The Xilinx LogiCORE™ IP Processor System Reset Module core provides customized resets for an entire processor system, including the processor, the interconnect and peripherals. The core allows customers to tailor their designs to suit their application by setting certain parameters to enable/disable features. The features of designs are discussed in the Design Parameters section.

Features

- Supports asynchronous external reset input which is synchronized with clock
- Both the external and auxiliary reset inputs are selectable as active-High or active-Low
- Selectable minimum pulse width for reset inputs to be recognized
- DCM Locked input
- Power On Reset generation
- Parameterized active-Low reset signal generation for core and for interconnect
- Sequencing of reset signals coming out of reset:

  a. Bus structures come out of reset (Interconnect and bridge)

  b. Peripherals come out of reset 16 clock cycles later (UART, SPI, IIC)

  c. The MicroBlaze™ processor comes out of reset 16 clock cycles after the peripherals

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ Families, UltraScale™ Architecture, Zynq®-7000, 7 Series
Supported User Interfaces	N/A
Resources	See Table 2-1 to Table 2-3.
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	Not Provided
Simulation Model	Not Provided
Supported S/W Driver	N/A
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	
Provided by Xilinx Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see Vivado IP catalog.
2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The Processing System Reset is a soft IP that provides a mechanism to handle the reset conditions for a given system. The core handles numerous reset conditions at the input and generates appropriate resets at the output. This core generates the resets based upon external or internal reset conditions.

Feature Summary

- Both the external and auxiliary reset inputs are selectable as active-High or active-Low. See Chapter 4, Customizing and Generating the Core for details.
- DCM Locked input. See Processor System Reset Module Circuit Description in Chapter 2 for details.
- Power On Reset generation. The Power On Reset condition causes all the reset outputs to become active within the first two clocks of a power up and remain active for 16 clocks. See Processor System Reset Module Circuit Description in Chapter 2 for details.
- Parameterized active-Low reset signal generation for core and for interconnect. See the Active-Low Reset option in Chapter 4, Customizing and Generating the Core.
- Sequencing of reset signals coming out of reset:

  a. Bus structures come out of reset (Interconnect and bridge)

  b. Peripherals come out of reset 16 clock cycles later (UART, SPI, IIC)

  c. The MicroBlaze™ processor comes out of reset 16 clock cycles after the peripherals

  See Processor System Reset Module Circuit Description in Chapter 2 for details.

Licensing and Ordering Information

This Xilinx LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

The Processor Reset Module block diagram is shown in Figure 2-1.

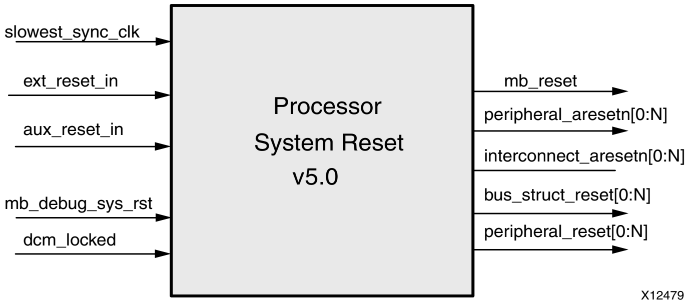


Figure 2-1: **Block Diagram for the Processor System Reset Module**

Processor System Reset Module Circuit Description

The core has five input signals and five output signals. You can set eight generics. Additional outputs can be generated through the use of the generics, **No. of Bus Reset (active-High)**, **No. of Peripheral Reset (active-High)**, **No. of Interconnect Reset (active-Low)**, and **No. of Peripheral Reset (active-Low)**. Figure 2-3 shows the Processor System Reset Module timing when an `ext_reset_in` occurs. The timing diagram is identical for an occurrence of **No. of Bus Reset (active-High)**. For this example **No. of Bus Reset (active-High)** is set to 5 and **External Reset Active Polarity** is set to 0, active-Low.

Core Functionality

The core receives two parameters:

- External Reset Active Window Width
- Auxiliary Reset Active Window Width

These parameters are used to set the minimum width of the external and auxiliary reset signals with respect to the `slowest_sync_clk` to allow the change to be detected.

For example, if **External Reset Active Window Width** is set to 5, then the *ext_reset_in* must become active and stay active for at least five clock cycles before a reset is initiated.

IMPORTANT: ***External Reset Active Window Width*** *sets the number of clock cycles. A change on* aux_reset_in *must be before the change is detected and used by the Processor System Reset Module.* aux_reset_in *performs exactly the same way as* ext_reset_in.

There is a one or two clock latency caused by the meta-stability circuit. Because ext_reset_in does not have to be synchronous with the input clock, the exact number of clocks cannot be determined. The reset becomes active in six or seven clocks after the input has gone active and stays active for five clocks. After ext_reset_in has gone inactive for five clocks the sequencing to come out of reset begins. If, during the sequencing, ext_reset_in goes active for five or more clocks all outputs become active again.

External Reset Active Polarity is used to set the logic level for which ext_reset_in causes a reset. If this generic is set to 1, when ext_reset_in is High on a rising edge of clock, a reset is initiated.

Auxiliary Reset Active Polarity is used to set the logic level for which aux_reset_in causes a reset. If this generic is set to 0, when aux_reset_in is Low, a reset is initiated.

No. of Bus Reset (active-High) and **No. of Interconnect Reset (active-Low)** are used to generate additional bus_struct_reset and Interconnect Structure Reset signals. This helps with signal loading and routing.

TIP: *In general, each bus can have its own* bus_struct_reset (interconnect_aresetn *in case of interconnect) signal.*

For example, if a system has one AXI4 interface then No. of Bus Reset (active-High) can be set to one. The bus_struct_reset output(s) should reset the arbiter(s) and bridges located on the bus. The same explanation applies to the number of Interconnect instances present in the system.

IMPORTANT: *The reset signal polarity for interconnect instances is active-Low when asserted.*

No. of Peripheral Reset (active-High) and **No. of Peripheral Reset (active-Low)** are used to generate additional peripheral_reset (active-High) and peripheral_aresetn (active-Low) signals. This helps with signal loading and routing. In general every peripheral can have its own peripheral_reset signal (peripheral_aresetn in case of peripherals connected to interconnect in a system).

For example, if there is one ATM, two UARTs, one 10/100 Ethernet controller and one IIC on the AXI4, then No. of Peripheral Reset (active-High) can be set to five. However, the **No. of Peripheral Reset (active-High)** can be set to one and all peripheral resets can be driven by the same output. The same explanation applies to the number of peripherals connected to Interconnect instances present in the system.

IMPORTANT: *The reset signal polarity for peripherals connected to interconnects are active-Low when asserted.*

mb_debug_sys_rst is an input signal that performs the same type of reset as ext_reset_in. The width of this signal complies to the same width requirement as for ext_reset_in, set by the **External Reset Active Window Width** parameter. The signal mb_debug_sys_rst is always active-High, (unaffected by the parameter **External Reset Active Polarity**). Normally this signal is connected to the Microprocessor Debug Module, MDM.

The signal dcm_locked is an input to the reset module. If the system does not use any DCMs this input should be tied High. If the system uses one DCM to generate system clocks, the output from the DCM should be connected to the input on the reset module.

IMPORTANT: *If dcm_locked is deasserted in between the normal operation, the Processor System Reset core asserts all of the reset outputs.*

If the system contains more than one DCM to generate system clocks, the DCM output that achieves lock last should be connected to the input.

The slowest_sync_clk input should be connected to the slowest synchronous clock used in the system. This is typically the AXI4-Lite interface clock, but could be any of the bus or CPU clocks.

All outputs go active on the same edge of the clock. However, there is a sequencing that occurs when releasing the reset signal.

- The first reset signals to go inactive are the bus_struct_reset and interconnect_aresetn.
- 16 clocks later peripheral_reset and peripheral_aresetn go inactive.
- 16 clocks later mb_reset goes inactive. Now all the resets are inactive and processing can begin.

There are two generics that decide the vector width of active-Low reset output signals. The parameter **No. of Interconnect Reset (active-Low)** and **No. of Peripheral Reset (active-Low)** decide the width of active-Low reset signals. The signal interconnect_aresetn provides the active-Low reset to interconnect and peripheral_aresetn provides the active-Low reset to peripherals. The width of interconnect_aresetn is similar to the bus_struct_reset signal width. The width of peripheral_aresetn is similar to the peripheral_reset signal width. The active-Low resets are mainly targeted for AXI-based peripherals or the peripherals which need an active-Low reset input. An example of using the Processor System Reset Module core is shown in Figure 2-2.

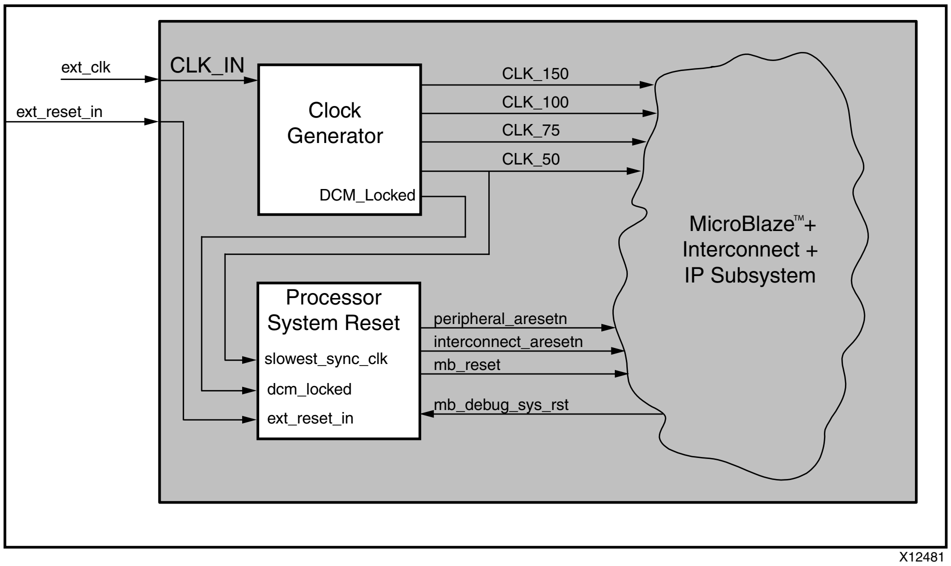


Figure 2-2: **Example of the Processor System Reset Module Core**

Figure 2-3 shows the core timing behavior. The Power On Reset condition causes all the reset outputs to become active within the first two clocks of a power up and remain active for 16 clocks. The resets then begin the sequencing as shown in Figure 2-3.

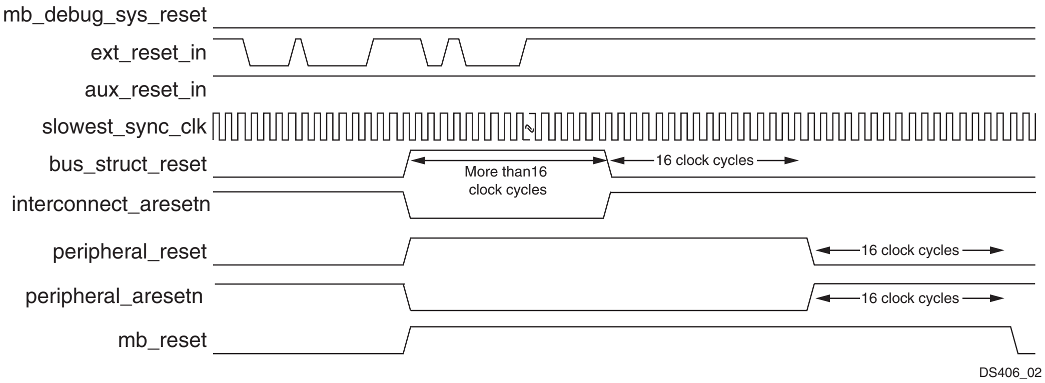


Figure 2-3: **Processor System Reset Module - Ext_Reset_In (active-Low)**

Performance

There is no information currently provided for this core.

Resource Utilization

Note: UltraScale™ architecture results are expected to be similar to 7 series device results.

Because the Processor System Reset IP core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the core is combined with other designs in the system, the utilization of FPGA resources and core timing vary from the results reported here.

The values reported in this section were generated using the Vivado Integrated Design Environment (IDE).

Virtex-7 Devices

The Processor System Reset IP core resource utilization for various parameter combinations measured with a Virtex®-7 FPGA as the target device is shown in Table 2-1. The utilization figures should be considered as reference only.

Table 2-1: **Device Utilization – Virtex-7 FPGAs**

Benchmark	Value
External Reset Active Window Width	15
Auxiliary Reset Active Window Width	16
Number of Peripheral Reset (Active-Low)	15
Number of Bus Reset (Active-High)	8
LUTs	30
FFs	62
LUT-FF Pairs	49
Slices	19
F_{Max} (MHz)	704

Kintex-7 Devices

Resources required for the Processor System Reset core have been estimated for the Kintex®-7 FPGA (Table 2-2). The utilization for the Kintex-7 based Zynq®-7000 family is expected to be the same as for the pure Kintex-7 device shown in Table 2-2.

Table 2-2: **Device Utilization – Kintex-7 FPGAs**

Benchmark	Value
External Reset Active Window Width	14
Auxiliary Reset Active Window Width	16
Number of Peripheral Reset (Active-Low)	16
Number of Bus Reset (Active-High)	7
LUTs	30
FFs	61
LUT-FF Pairs	53
Slices	21
F_{Max} (MHz)	704

Artix-7 Devices

The utilization for the Artix®-7 based Zynq-7000 family is expected to be same as for the pure Artix-7 device shown in Table 2-3.

Table 2-3: **Device Utilization – Artix-7 FPGAs**

Benchmark	Value
External Reset Active Window Width	14
Auxiliary Reset Active Window Width	14
Number of Peripheral Reset (Active-Low)	16
Number of Bus Reset (Active-High)	8
LUTs	29
FFs	59
LUT-FF Pairs	48
Slices	18
F_{Max} (MHz)	508

Port Descriptions

The I/O signals for the Processor System Reset Module are listed in Table 2-4.

Table 2-4: **Processor System Reset Module I/O Signals**

Port	Signal Name	Interface	I/O	Description
P1	slowest_sync_clk	System	I	Slowest Synchronous Clock. Typically AXI4-Lite clock.
P2	ext_reset_in	System	I	External Reset Input. Active-High or Low based upon the generic Ext Reset Active Polarity.
P3	mb_debug_sys_rst	System	I	MDM reset input. Always active-High, minimum width defined by parameter External Reset Active Window Width.
P4	aux_reset_in	System	I	Auxiliary Reset Input. Active-High or Low based upon the generic Auxiliary Reset Active Polarity.
P5	dcm_locked	System	I	DCM Lock signal.
P6	mb_reset	System	O	MB core reset. Active-High.
P7	bus_struct_reset[0 to No. of Bus Reset (active-High) - 1][1]	System	O	Bus Structures reset - for example, arbiters for bridges. Active-High
P8	peripheral_reset[0 to No. of Peripheral Reset (active-High) - 1][2]	System	O	Peripheral reset is for all peripherals attached to any bus that is synchronous with the slowest_sync_clk. Active-High.
P9	interconnect_aresetn[0 to No. of Interconnect Reset (active-Low) - 1][1]	System	O	Interconnect_aresetn reset, for example, interconnects with active-Low reset inputs.
P10	peripheral_aresetn [0 to No. of Peripheral Reset (active-Low) - 1][2]	System	O	The signal peripheral_aresetn is for all peripherals attached to interconnect that is synchronous with the slowest_sync_clk. Active-Low.

Notes:

1. To help equalize loading on this signal, there can be from 1 to 8 copies generated with each copy being individually registered through a D-flip flop. In general each unique bus should receive a different copy of this signal.
2. To help equalize loading on this signal, there can be from 1 to 16 copies generated with each copy being individually registered through a D-flip flop. In general each peripheral should receive a different copy of this signal.

Chapter 3

Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the core.

Clocking

The core should be connected to the slowest operating clock of the system.

Resets

The external reset input port should be connected to the external reset pin, while the auxiliary reset input should be connected to the internal source for reset. The MicroBlaze™ Debug Reset should be connected to the debug module of the system. In the case where the auxiliary reset input and/or MicroBlaze Debug Reset ports are un-connected, these ports are driven with the deasserted values.

Chapter 4

Customizing and Generating the Core

This chapter includes information about using the Vivado® Design Suite to customize and generate the core.

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 1] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value you can run the `validate_bd_design` command in the Tcl Console.

Vivado Integrated Design Environment

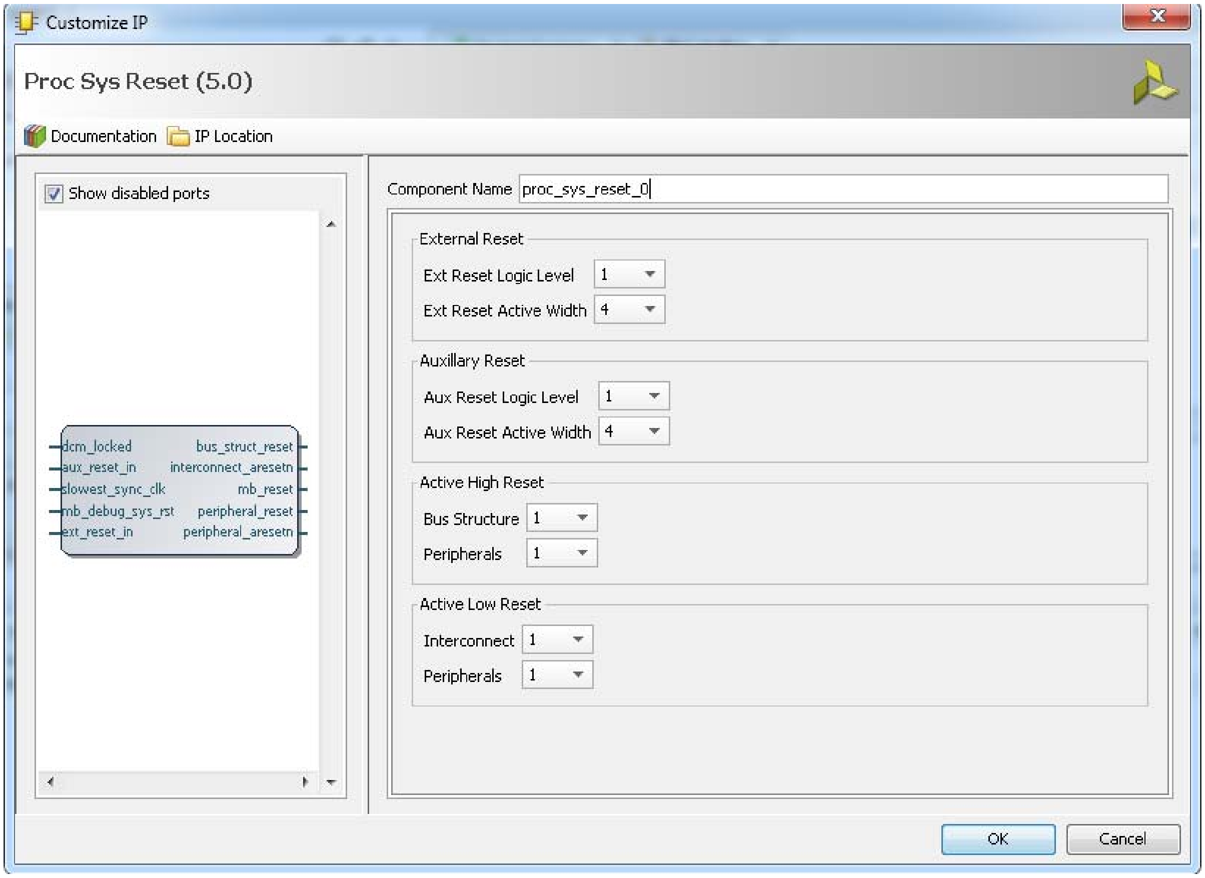
You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

1. Select the IP from the IP catalog.
2. Double-click the selected IP, or select the **Customize IP** command from the toolbar or popup menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3].

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). This layout might vary from the current version.

Figure 4-1 shows the Vivado IDE.



Customize IP

Proc Sys Reset (5.0)

☑ Show disabled ports

Component Name: proc_sys_reset_0

External Reset
- Ext Reset Logic Level: 1
- Ext Reset Active Width: 4

Auxillary Reset
- Aux Reset Logic Level: 1
- Aux Reset Active Width: 4

Active High Reset
- Bus Structure: 1
- Peripherals: 1

Active Low Reset
- Interconnect: 1
- Peripherals: 1

Figure 4-1: **Vivado Integrated Design Environment**

The Vivado IDE provides the following options.

- **External Reset**
  - **External Reset Logic Level**. This option determines if the external reset pin is active-Low or active-High.
  - **External Reset Active Width**. This option ensures that if the reset pin is active for the selected duration, it is treated as a legal reset and the core generates the other reset pins.
- **Auxiliary Reset**
  - **Auxiliary Reset Logic Level**. This option determines if the auxiliary reset pin is active-Low or active-High.
  - **Auxiliary Reset Active Width**. This option ensures that if the auxiliary reset pin is active for selected duration, it is treated as a legal reset and the core generates the other reset pins.

- **Active High Reset**
  - **Bus Structure Reset**. This option provides the number of resets generated for the bus interface. The reset is active-High polarity.
  - **Peripherals**. This option provides the number of resets generated for peripherals. The reset is active-High polarity.
- **Active Low Reset**
  - **Interconnect**. This option provides the number of resets generated for the Interconnect interface. The reset is active-Low polarity.
  - **Peripherals**. This option provides the number of resets generated for peripherals. The reset is active-Low polarity.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Chapter 5

Constraining the Core

There are no constraints available for this core.

Chapter 6

Simulation

For comprehensive information about Vivado® simulation components, as well as information about using supported third party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4].

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado® Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Example Design

This chapter contains information about the provided example design in the Vivado® Design Suite environment.

The top-level example design for the Processor System Reset Module core is located here:

```
<project_name>/<project_name>.srcs/sources_1/ip/<component_name>/
<component_name>example_design/<component_name>_exdes.vhd
```

The example design performs the following tasks:

- Instantiates the IP top-level file and generates various reset inputs for the IP.
- Takes in the Reset widths options from the Vivado Integrated Design Environment (IDE), and generates input resets with a width greater-than and less-than the user defined values to perform negative testing.
- Implements Checker logic to test the widths of the generated resets and flag error in cases of discrepancies.

The example design has Timeout logic to ensure simulation/board test is not run forever.

Figure 8-1 shows the Processor System Reset Module example design block diagram.

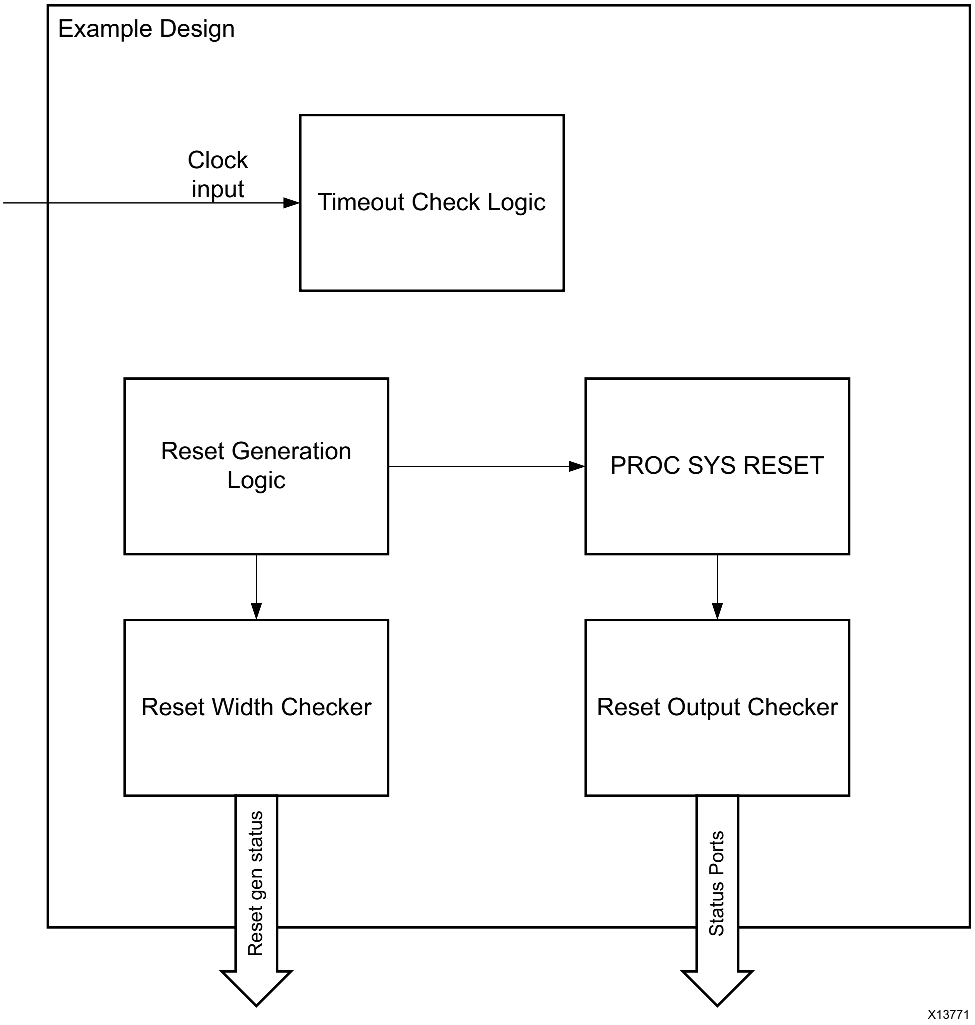


Figure 8-1: **Processor System Reset Module Example Design Block Diagram**

Implementing the Example Design

After following the steps described in Chapter 4, Customizing and Generating the Core to generate the core, implement the example design as follows:

1. Right-click the core in the Hierarchy window, and select **Open IP Example Design**.
2. A new window pops up, asking you to specify a directory for the example design. Select a new directory or keep the default directory.
3. A new project is automatically created in the selected directory and it is opened in a new Vivado window.
4. In the Flow Navigator (left-side pane), click **Run Implementation** and follow the directions.

Example Design Directory Structure

This directory and its subdirectories contain all the source files that are required to create the Processor System Reset Module example design.

Table 8-1 shows the files delivered in the example design directory. It contains the generated example design top files.

Table 8-1: **Example Design Directory**

Name	Description
<project_name>/<project_name>.srcs/sources_1/ip/ topdirectory	Top-level project directory; name is user-defined
<project_name>/<project_name>.srcs/sources_1/ip/ <component_name>/example design	VHDL example design files.
<project_name>/<project_name>.srcs/sources_1/ip/ <component_name>/simulation	VHDL Test Bench file.

Table 8-2 shows the files delivered in the `<project_name>/<project_name>.srcs/sources_1/ip/` directory.

Table 8-2: **Project Files**

Name	Description
Synth/<component name>.v|vhd	Synthesis wrapper generated by the Vivado tool.
Sim/<component name>.v|vhd	Simulation wrapper generated by the Vivado tool.
<component_name>.xci	Vivado tools project-specific option file; can be used as an input to the Vivado tools.
<component_name>.vho|veo	VHDL or Verilog instantiation template.
<component_name>_ooc.xdc	Out of Context constraints for IP.

Table 8-3 shows the files delivered in the `<project_name>/<project_name>.srcs/sources_1/ip/<component_name>` provided with the core.

Table 8-3: **Example Design Files**

Name	Description
<component name>_exdes.vhd	Example design top file for synthesis.

Table 8-4 shows the file delivered in the simulation directory provided with this core.

Table 8-4: **Simulation Directory**

Name	Description
<component name>_top_tb.vhd	Test bench file for simulation.

Simulating the Example Design

Using the Processor System Reset Module example design (delivered as part of the Processor System Reset Module), you can quickly simulate and observe the behavior of the Processor System Reset Module.

Setting Up the Simulation

The Xilinx simulation libraries must be mapped into the simulator. If the libraries are not set for your environment, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4] for assistance compiling Xilinx simulation models and setting up the simulator environment. To switch simulators, click **Simulation Settings** in the Flow Navigator (left pane). In the Simulation options list, change **Target Simulator**.

Simulation Results

The simulation script compiles the Processor System Reset Module example design and supporting simulation files. It then runs the simulation and checks to ensure that it completed successfully.

If the test passes, then the following message is displayed:

```
Test Completed Successfully
```

If the test fails or does not complete, then the following message is displayed:

```
Test Failed!! Test Timed Out.
```

Test Bench

This chapter contains information about the provided test bench in the Vivado® Design Suite environment.

The demonstration test bench for the Processor System Reset Module core is located here:

```
<project_name>/<project_name>.srcs/sources_1/ip/<component_name>/
<component_name>/simulation/<component_name>_top_tb.vhd
```

Figure 9-1 shows a top-level view of the example design test bench.

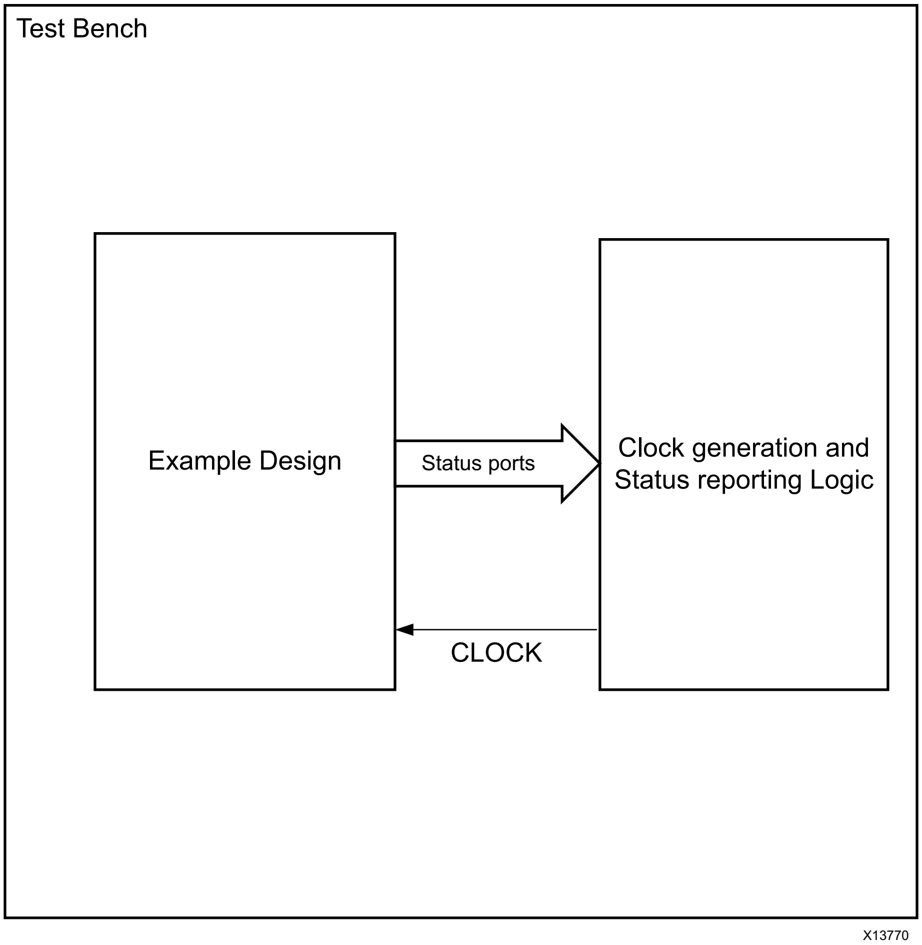


Figure 9-1: **Processor System Reset Module Example Design Test Bench**

The demonstration test bench performs the following tasks:

- Generates clock input for the example design, for timeout logic, and reset width calculation logic
- Takes the status from example design and reports to the user using report statements.

Appendix A

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® Design Suite to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

For information on migrating to the Vivado Design Suite, see the *Vivado Design Suite Migration Methodology Guide* (UG911) [Ref 5].

Upgrading in the Vivado Design Suite

There are no port or parameter changes.

Appendix B

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the Processor System Reset Module, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for opening a Technical Support WebCase.

Documentation

This product guide is the main document associated with the Processor System Reset Module. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can also be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as:

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the Processor System Reset Module

AR: 54433

Technical Support

Xilinx provides technical support in the Xilinx Support web page for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Note: Access to WebCase is not available in all cases. Login to the WebCase tool to see your specific support options.

Debug Tools

Vivado Design Suite Debug Feature

Vivado® lab tools insert logic analyzer and virtual I/O cores directly into your design. Vivado lab tools allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature represents the functionality in the Vivado Integrated Design Environment (IDE) that is used for logic debugging and validation of a design running in Xilinx devices in hardware.

The Vivado logic analyzer is used to interact with the logic debug LogiCORE IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 6].

Debugging General IP Behavior

The Processor System Reset Module does not have any AXI interface. It has one clock and three reset inputs to deal with the system reset generation.

The following checks should be done to make sure the IP is behaving correctly:

- The IP should be connected to the slowest of the system clocks.
- The DCM lock of the final DCM instance should be connected to the lock input pin of the core.
- The external reset is assumed to be generated from the board, which should have initial RC filtering logic to dampen the reset key bouncing.
- The other two reset signals are internal to the system.
- Make sure that the reset active polarity and proper width (for filtering the reset glitches) is properly selected when the core is customized in the Vivado IDE.
- The reset pulse width should be at the minimum or more of the active width selected in Vivado IDE core customization.
- After the reset is active, the core asserts all the reset output signals to active state.
- You should connect the proper resets to their cores.

- The interconnect (bus structure) reset is deasserted first, then peripheral resets are deasserted, and at the end the processor reset signal is deasserted.

Appendix C

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm

References

These documents provide supplemental material useful with this product guide:

1. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
2. *Vivado Design Suite User Guide: Designing with IP* (UG896)
3. *Vivado Design Suite User Guide: Getting Started* (UG910)
4. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
5. *Vivado Design Suite Migration Methodology Guide* (UG911)
6. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/18/2015	5.0	• Added support for UltraScale+ families. • Added dcm_locked Important note in Core Functionality section.
12/18/2013	5.0	Added UltraScale support.
10/02/2013	5.0	Re-release of the core v5.0 document, with the following changes: • Added core support in Vivado IP Integrator. • Added information related to the core design example, and test bench. • Added Simulation, and Synthesis and Implementation chapters. • Updated Debugging General IP Behavior section.
03/20/2013	5.0	Initial Xilinx release of this document in the product guide format. This new guide is based on ds406.

Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.