# **Control Interfaces and Processing System v2.1**

# LogiCORE IP Product Guide

Vivado Design Suite

PG352 (v2.1) December 4, 2020





# Table of Contents

Chapter 1: Introduction	4
Features	4
IP Facts	5
Chapter 2: Overview	6
Navigating Content by Design Process	6
Core Overview	6
Licensing and Ordering	6
Chapter 3: Product Specification	8
I/O Peripheral and Flash Memory Controllers	8
Chapter 4: Design Flow Steps	10
Customizing and Generating the Core	10
Automation	11
CPM4	14
Device Integrity	15
PS-NoC Interfaces	21
Clocking Configuration	27
I/O Configuration	
PL-PS Configuration	35
Programmable Logic Interrupts	
Debug Settings	
Boot Mode	44
Unsupported Features	45
Appendix A: Debugging	46
Finding Help on Xilinx.com	46
Debug Tools	47
Appendix B: Additional Resources and Legal Notices	49
Xilinx Resources	49



Documentation Navigator and Design Hubs	49
References	49
Revision History	
Please Read: Important Legal Notices	50





# Chapter 1

# Introduction

The Xilinx<sup>®</sup> Versal<sup>™</sup> platform Control, Interfaces, and Processing System IP is the software interface around the Versal processing system. The Versal family consists of a system-on-chip (SoC) style integrated processing system (PS) and a programmable logic (PL) unit, NoC, and AI Engine providing an extensible and flexible SoC solution on a single die.

### Features

- Enable/Disable I/O peripherals (IOP)
- Enable/Disable AXI Interfaces
- Multiplexed I/O (MIO) configuration
- Extended multiplexed I/Os (EMIO)
- PL clocks, interrupts, and resets
- Internal clocking (PMC/LPD/FPD)
- Generation of system level configuration registers (SLCRs)
- Enable/Disable PS to/from NoC Interface
- CCIX and PCIe<sup>®</sup> configuration
- SYSMON configuration
- Debug configuration



## **IP Facts**

LogiCORE™ IP Facts Table					
	Core Specifics				
Supported Device Family	Versal™ ACAP				
Supported User Interfaces	AXI4, AXI4-Lite, AXI4-Stream, Native, and NoC				
	Provided with Core				
Design Files	Verilog				
Example Design	Refer block automation for DDR/CPM reference designs				
Test Bench	N/A				
Constraints File	N/A				
Simulation Model	N/A				
Supported S/W Driver N/A					
Tested Design Flows					
Design Entry	Vivado® Design Suite				
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.				
Synthesis	Vivado Synthesis				
	Support				
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775				
	Xilinx Support web page				

#### Notes:

1. For a complete list of supported devices, see the Vivado<sup>®</sup> IP catalog.

2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.



Chapter 2



# Overview

# Navigating Content by Design Process

Xilinx<sup>®</sup> documentation is organized around a set of standard design processes to help you find relevant content for your current development task. This document covers the following design processes:

- Hardware, IP, and Platform Development: Creating the PL IP blocks for the hardware platform, creating PL kernels, subsystem functional simulation, and evaluating the Vivado<sup>®</sup> timing, resource use, and power closure. Also involves developing the hardware platform for system integration. Topics in this document that apply to this design process include:
  - I/O Peripheral and Flash Memory Controllers
  - Chapter 4: Design Flow Steps
  - Clocking

### **Core Overview**

The Control Interfaces and Processing System IP core instantiates, boots, and configures the processing system section of the Xilinx<sup>®</sup> Versal<sup>™</sup> platform.

## **Licensing and Ordering**

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

*Note*: To verify that you need a license, check the License column of the IP Catalog. Included means that a license is included with the Vivado<sup>®</sup> Design Suite; Purchase means that you have to purchase a license to use the core.



Information about other Xilinx<sup>®</sup> LogiCORE<sup>™</sup> IP modules is available at the Xilinx Intellectual Property page. For information about pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.





# Chapter 3

# **Product Specification**

The functional block diagram of the core is shown in the following figure.



#### Figure 1: Core Block Diagram

# I/O Peripheral and Flash Memory Controllers

The I/O Peripheral and Flash Memory Controllers include the following.



- Quad Serial Peripheral Interface (QSPI) flash memory
- Octal Serial Peripheral Interface (OSPI) flash memory
- UART
- I2C
- Serial Peripheral Interface (SPI) flash memory
- SD/eMMC
- General purpose I/O (GPIO)
- Controller Area Network Flexible Data rates (CAN-FD)
- USB 2.0
- Gigabit Ethernet MAC (GEM)

**Note:** The interfaces for these I/O peripherals (IOPs) can be routed to MIO ports and the extended multiplexed I/O (EMIO) interfaces as described in the *Versal ACAP Technical Reference Manual* (AM011).

- PMC domain peripheral
  - 。QSPI
  - 。 OSPI
  - 。 2xSD/eMMC
  - 。 I2C
  - 。 SelectMAP (SMAP)
  - 。 GPIOs
- Low power domain (LPD) peripherals available in PS:
  - 。 2 X Gigabit Ethernet
  - 。 1 X USB2.0
  - 。 2 X SPI
  - 。 2 X CAN
  - 。 2 X I2C
  - 。 2 X UART
  - 。 GPIOs



# Chapter 4

# **Design Flow Steps**

This section describes customizing and generating the core, constraining the core, and the simulation, synthesis, and implementation steps that are specific to this IP core. More detailed information about the standard Vivado<sup>®</sup> design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator (UG994)
- Vivado Design Suite User Guide: Designing with IP (UG896)
- Vivado Design Suite User Guide: Getting Started (UG910)
- Vivado Design Suite User Guide: Logic Simulation (UG900)

### **Customizing and Generating the Core**

This section includes information about using Xilinx<sup>®</sup> tools to customize and generate the core in the Vivado<sup>®</sup> Design Suite.

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) 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, run the validate\_bd\_design command in the Tcl console.

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 rightclick menu.

For details, see the Vivado Design Suite User Guide: Designing with IP (UG896) and the Vivado Design Suite User Guide: Getting Started (UG910).

Figures in this chapter are illustrations of the Vivado IDE. The layout depicted here might vary from the current version.



### **CIPS IP Core Block Design**

You can customize the Control Interfaces and Processing System IP core by clicking on the **CIPS IP**. The block diagram on selecting the customization is shown in the following figure.



#### Figure 2: CIPS IP Core Block Design

### Automation

The CIPS IP provides optional assistance to the designer such as Block and Board Automation. Block Automation provides an initial configuration and connects to additional related IP blocks. Board Automation applies a specific configuration preset to CIPS IP when a board part is chosen and has a preset.

In addition, other IPs may provide Connection Automation for additional peripheral/connectivity to be connected to CIPS IP. Following are the automation limitations:

- Block, board, and connection automation are independent entities in the IP integrator. Using multiple automations may cause conflicts.
- Connection Automation may not recognize hardened interfaces.



### **Block Automation**

Vivado<sup>®</sup> supports the Block Automation for Control, Interfaces, and Processing System IP to aid in integrating it into the larger design. After adding the CIPS IP to the block diagram, the block automation banner pops up. Click **Run Block Automation** to open the block automation page.



Figure 3: Block Automation

Block automation supports the following options:

- Memory Controller: You can select x1/x2/x4 interleaved DDR4 DRAM controllers and a new/ existing NoC to be connected to the CIPS IP core.
- PL clocks/PL resets: You can select 0-4 PL clocks and 0-4 reset signals which are exposed to PL.
- **CPM4**: You can configure the PCIe0/1 controllers in CPM4 with different use modes (PCIE/DMA/CCIX), different port types (EP/RC), and different supported lane widths (X1/X2/X4/X8/X16). In the DMA mode, PCIe0/1 controllers are connected to PMC slave port over NoC.





X ¢	Description	
] All Automation (1 out of 1 selected) ✓ ♥ versal_cips_0	The Control, Interface, and Processing System block automation wizard assists with the generation switch block. It will also assist with the connections for DDR4, PCIe, and programmable fabric inter Instance: /versal_cips_0 Options	on of a NOC erface ports.
	PS/PMC Memory Controller (DDR4) PL Clocks None PL Resets None CPM4	
	PCIe A0 PCIe A1	
	Mode None 🗸 Mode None 🗸	
	Lane Width None 🗸 Lane Width None 🗸	
	Port Type Endpoint Device 🗸 Port Type Endpoint Device 🗸	
	ΑΧΊ ΝοC	
	Configure NoC Add new AXI NoC 👻	

#### Figure 4: Run Block Automation

When you click OK, a validate ready design is provided with input requirements.

*Note*: The block automation banner disappears when CIPS IP instance configuration is updated/changed. Thus, it must be the first configuration step if intended to be used..

### **Board Automation**

When you create a Vivado<sup>®</sup> project targeting a board instead of a specific device, a board preset may be available to initialize the CIPS IP core with board-specific settings.

After instantiating the CIPS IP in the block design, open CIPS GUI to view the Board page as shown in the following figure. Select **cips fixed io** option to apply board preset to CIPS core. A preset is not applied if you select the **Custom** option and you have to go to the IO configuration page to enable peripherals. Board presets may also be applied by dragging/dropping from the Vivado Board tab onto an empty space on the canvas or onto an existing CIPS instance.



Configuration Options Board Associate IP Interface with board interface IP Interface Board Interface Configuration Options Board Definition Def	
Configuration Options         Associate IP Interface with board interface           Board         IP Interface         Board Interface	
Board IP Interface Board Interface	
DC Custom	
Home PS Custom	,
Boot Mode Custom	
Debug Clear Board Parameters cips fixed io	
> PS-PMC	
© CPM4	
Device Integrity	
Denice integrity	
Sysmon Configuration	
Sysmon Configuration XIISEM Library Configuration	
> PS-PMC > CPM4	

#### *Figure 5:* **Board Automation**

## CPM4

The Control, Interfaces, and Processing System IP core has no internal PCIe<sup>®</sup> module. Instead, it has direct interfaces to CPM4 (CCIX-PCie module), which includes a complete subsystem for PCIe. CPM4 is an independent subsystem that exists in some Versal<sup>™</sup> devices.

Documentation 🛛 🚍 IP Location					
omponent Name versal_cips_0					
Configuration Options	CPM4 Basic Configuration				
Home > Boot Mode	PCIe Controller 0				
> Debug > PS-PMC ~ CPM4	PCIe Controller 0 Modes	None 🗸	Lane Width	NONE 🗸	
CPM4 Basic Configuration	PCIe Controller 1				
CPM4 PCIE Controller 0 Configuration CPM4 PCIE Controller 1 Configuration CPM4 CCIX Configuration	PCIe Controller 1 Modes CPM4 Interfaces	None 🗸	Lane Width	NONE 🗸	
> Device Integrity	CPM to NoC 0 CF	PM to NoC 1	NoC to CPM		

#### Figure 6: CPM4 Basic Configuration

**Note:** For more information on CPM module, see Versal ACAP CPM Mode for PCI Express Product Guide (PG346), Versal ACAP CPM DMA and Bridge Mode for PCI Express Product Guide (PG347), and Versal ACAP CPM CCIX Architecture Manual (AM016).





# **Device Integrity**

Device integrity feature enables you to monitor and respond to the system operating conditions and exceptional events. Many of these features are broadly applicable to many designs, while others are provided in support of meeting stringent reliability, security, and safety requirements. Device integrity has three modules namely, Sysmon, XilSEM Library, and Tamper configurations as described in subsequent sections.

### **System Monitor**

The System Monitor (SYSMON) enables monitoring of the physical environment both within the Versal<sup>™</sup> ACAP itself and also within the wider system using the external inputs. It is used to ensure that both the Versal ACAP and the overall system operate in a safe, secure, and reliable way. The SYSMON provides the customer with a digital measure of the temperature and applied voltage supplies as well as off-chip voltage measurements within the broader system context. Off-chip its main uses are for board level monitoring of supply voltages/currents.

The Control, Interfaces, and Processing System IP core can enable the following measurements:

- On-chip Supply Monitor
- Temperature Measurements
- External Supply Measurements

The Basic Configuration tab has Default and Custom options, as shown in the following figure.





ocumentation 📄 IP Location				
mponent Name versal_cips_0				
onfiguration Options	Basic Configuration On Chip St	upply Monitor	Temperature	External Supply Monitor
Board	Common Configuration Template	ustom		×
is-PMC PM4 Pevice Integrity Sysmon Configuration XIISEM Library Configuration Tamper	What Interface Do you Want To Use De-restrict PMBus Commands PMBUS Address I2C IO Reference Used by SYSMON Custom	None 0x0 PS MIO 23 2 Internal	* 5 * *	
	Voltage Averaging			
	Number Of Samples for Volta	ige Averaging 🗌	None 🗸	

#### Figure 7: SYSMON Basic Configuration

The default window has different preset options to provide a common starting point for typical SYSMON usage. Each measurement has associated threshold levels which control alarm assertion. The alarms are enabled by default. For external access to SYSMON measurements, I2C, and PMBus interfaces are supported.

The default window also provides the option for setting the voltage averaging levels of 2, 4, 8, and 16.

#### **On-Chip Supply Monitor**

The On-chip supply monitor tab supports different types of voltage measurements including customer supply voltages and customer dedicated pad voltages.

The On Chip Supply Monitor configuration is shown in the following figure.



Documentation 🛯 🖨 IP Location								
omponent Name versal_cips_0								
Configuration Options	Basic Configuration	On Chip Supply Monitor	Temperature	External Supr	ly Monitor			
Home > Boot Mode	← Q ± ¢							
Debug	Search: Q-							
PS-PMC CPM4	Name	Enable	Avg Enable	Alarm Enable	ADC Mode	Threshold Low	Threshold High	
<ul> <li>Device Integrity</li> </ul>	SUPPLY_VOLTAGE							
Sysmon Configuration	> VCC							
XilSEM Library Configuration	> VCCINT							
Tamper	> VCCAUX							
	V DEDICATED_PAD							
	> VP							
	V BANK_VOLTAGE							
	> VCCO							
	> VCC							
	> VCCAUX							
	> GTY AVIT							

#### Figure 8: On Chip Supply Monitor

The CIPS IP core allows you to enable averaging, define the ADC Mode (unipolar/bipolar), and enable alarms with user-defined upper and lower thresholds.

#### **Temperature Measurements**

The Temperature configuration tab configures device temperature monitoring options, including over-temperature shutdown. Alarms are all specified in window mode, where the alarm is asserted above the upper threshold or blow the lower threshold. You can configure the lower and upper temperature value based on the design requirements.

Figure 9:	Temperature	Configuration

ocumentation 🛛 🕞 IP Location		
mponent Name versal_cips_0		
onfiguration Options Board	Basic Configuration On Chip Supply Monitor Temperature External Supply Monitor	
Home Boat Mode Debug PS-PMC CPM4 Device Integrity Sysmon Configuration XIISEM Library Configuration	Device Over-Temperature         Over Temperature Lower (Celcius)         0/ser Temperature Upper (Celcius)         125         Cover Temperature Upper (Celcius)         125         Device Temperature Upper (Celcius)         125         Device Temperature Upper (Celcius)         125         Device Temperature Upper (Celcius)         Device Temperature Upper (Celcius)         125         Device Temperature Upper (Celcius)         Device Temperature Upper (Celcius) </td <td>the OT alarm Ilow</td>	the OT alarm Ilow
Tamper	User Temp Alarm Type       window       Active Thermal Management         User Temperature Lower (Celcius)       0       0         User Temperature Upper (Celcius)       125       0         • Window: Alarm is asserted when the temperature is between the upper and lower       • Hysteresis: Alarm is asserted when the temperature goes above the upper threshold and deasserted when it goes below the lower threshold.	thresholds,



#### **External Supply Measurements**

The Control, Interfaces, and Processing System IP core allows up to 16 pins to be selected for external supply measurements. All AUXIOs should be assigned to the same bank.

For example, if AUX\_IO\_P is LPD Bank MIO0 (VCCO\_502 rail), then AUX\_IO\_N should be assigned from LPD MIO Bank MIO (VCCO\_502 rail).

The AUXIOs supports PMC MIOs, LPD MIOs, and HDIOs. But PMC and LPD MIOs may have conflict based on PS-PMC IO Configuration page. To resolve the conflict, use other free available IOs.

The External Supply monitor configuration tab is shown in the following figure:

ocumentation 🛸 IP Location										
omponent Name versal_cips_0										
Configuration Options	Basic Configuration	On Chi	p Supply Monitor	Temperature	External Sup	olv	Monitor			
Board Home	+ Q ₹ \$	1								
Boot Mode     Debug	Search: Q-									
> PS-PMC	Channel	Enable	AUX_IO_P	AUX_IO_N	ADC MODE		Average_Enable	Alarm_Enable	Threshhold_Lower	Threshold_Upper
> CPM4	<ul> <li>Supply Monitors</li> </ul>	0							-	
Sysmon Configuration	vauxp0/vauxn0		PMC MIOD SOL ~	PMC MIOT SOL ~	1 V unipolar	Y			U	0
XIISEM Library Configuration	vauxp1/vauxn1		PMC MI00 50( ~	PMC MI01 50( ~	1 V unipolar	~			0	0
Tamper	vauxp2/vauxn2		PMC MIOD 50( $\sim$	PMC MI01 50( $\sim$	1 V unipolar	$\sim$			0	0
	vauxp3/vauxn3		PMC MIO0 50( $\sim$	PMC MI01 50( $ \lor$	1 V unipolar	v.			0	0
	vauxp4/vauxn4		PMC MI00 50( 😔	PMC MI01 50( ~	1 V unipolar	v			0	0
	vauxp5/vauxn5		PMC MI00 50( ~	PMC MI01 500 V	1 V unipolar	~			0	0
	vauxp6/vauxn6		PMC MI00 50( ~	PMC MI01 50( ~	1 V unipolar	~			0	0
	vauxp7/vauxn7		PMC MI00 50( ~	PMC MI01 50( ~	1 V unipolar	~			0	0
	vauxp8/vauxn8	0	PMC MI00 50( ~	PMC MI01 50( V	1 V unipolar	~			0	0
	vauxp9/vauxn9		PMC MI00 500 ~	PMC MI01 500 V	1 V unipolar	~			0	0
	vourpovolunio	0	PMC MICO 50/	PMC MIO1 500 ···	1.1/ unipolar				0	0
	vauxpro/vauxnic	0	PHC MICO 501 0	Phie Miel Str	1 v unipular	-			•	
	vauxp11/vauxn11	U	PMC MI00 50( V	PMC MIOT SOC V	1 V unipolar	~			U	U

#### Figure 10: SYSMON External Supply Monitor

### **XilSEM Library Configuration**

The Xilinx<sup>®</sup> soft error mitigation (XilSEM) library is a pre-configured, pre-verified solution to detect and optionally correct soft errors in Versal<sup>™</sup> ACAP configuration memory. A soft error is caused by ionizing radiation and is extremely uncommon in commercial terrestrial operating environments. While a soft error does not damage the device, it carries a small statistical possibility of transiently altering the device behavior.

The XilSEM library does not prevent soft errors; however, it provides a method to better manage possible system-level effect. Proper management of a soft error can increase reliability, availability, and reduces system maintenance and downtime. In most applications, soft errors can be ignored. The XilSEM library configuration options available through the CIPS IP core are shown in the following figure.



Figure	11: XilSEM	Library	y Configuration
--------	------------	---------	-----------------

omponent Name versal cips 0		
Configuration Options Board	Step 1 What Device Configuration Soft Errors Would You Like To Mitigate?	
Home Boot Mode Debug PS-PMC	Configuration RAM Soft error mitigation of user memory resources should be handled through design entry.	
CPM4 Device Integrity Sysmon Configuration	Step 2       What Do You Want To Do With Error(s)?       Detect & Correct	
XilSEM Library Configuration	Step 3	
lamper	Configuration RAM Options  Disable HW calculated ECC	8

In CIPS IP v2.1, you can choose to enable or disable the error detection and correction capabilities of the XilSEM library as shown in Figure 11: XilSEM Library Configuration. In applications that require soft error mitigation, see Versal ACAP System Software Developers Guide (UG1304) for additional information about the XilSEM library prior to configuring it for use.

### **Tamper Events/Response Configuration**

Tamper events are interrupts from a tamper monitoring function and user can select different responses to each tamper events.





#### Figure 12: Tamper Response System



Tamper monitoring system in CIPS generates interrupts as an when it detects below events.

- **Supply Glitch:** Whenever there is a glitch in power supply happens, then this event will get generated. Glitching the power supply (low or high) can cause insecurely designed state machines to skip states. Glitches are injected at various points in time and vary in pulse widths.
- Voltage Deviation: Both low and high voltage cause race conditions that can trip just about any type of circuity, this event would be generated when there is a Voltage deviation. Tamper monitoring system supports LPD, FPD, PMC, SOC, RAM, and Bank3 IO VCC voltage deviations.
- **Temperature Deviation:** When device temperature goes out of specification (high or low) then this event will be generated. This is commonly done in conjunction with a voltage attack. Both low and high temperature cause race conditions that can trip just about any type of circuity.
- **Debug (JTAG) toggle detect:** Debug interfaces attack the Silicon devices, most frequently this is the JTAG port but with the growing complexity of devices more advanced debug interfaces are becoming more prominent. This event would be generated when there is a toggle in debug interfaces.
- **Custom User (External MIO) event: :** This event is generated when Tamper monitoring system detects any interrupt (active high) on external MIO.
- **Tamper Register event:** When you directly trigger the tamper system by writing to its specific register, then this event would be generated.

CIPS has different responses to each of the above mentioned tamper events, which are described below.

• **BBRAM Zeroization:** When a tamper event is detected, it is required that the you can immediately erase the key stored in BBRAM. However, in high grade crypto applications, it is not sufficient to simply delete the key when done or when a tamper event is detected. It is required to be zeroized (erase + verify). CIPS provides this response for all the tamper events.



- Secure Lockdown: Upon detection of a tamper event, you want the system to go into some form of lockdown state. CIPS provides lockdown response for all the tamper events.
- Secure Lockdown (With IO Tri State): Some systems require a more severe response to a tamper event and even secure lockdown is not enough. It such cases it is necessary to also tristate all IO to the device. This makes it impossible for the adversary to gain any level of access to the device after a tamper event. CIPS provides lockdown with IO Tristate response for all the tamper events.
- **System Reset:** You may want to only Reset the system, upon receiving the tamper event. CIPS provides System Reset response for all the tamper events.
- **System Interrupt:** You may only want to know that the tamper event is occurred. Tamper response system generates an interrupt to system, upon receiving any tamper event.

You can select BBRAM Zeroization or Secure Lockdown or Secure Lockdown (With IO Tri State) or System Reset or System Interrupt for each Tamper Event as response in PCW.

ocumentation 🛛 🖨 IP Location				
omponent Name versal_cips_0				
Configuration Options	Tamper Configuration			
Home	Security Related Tamper Event	Zeroize BBRAM	Tamper Response	
> Boot Mode	✓ JTAG Toggle		SYS INTERRUPT	-
> PS-PMC	Temperature Alarm		SYS INTERRUPT	
> CPM4	Voltage Alarm For VCC PSLP		SEC LOCKDOWN	
<ul> <li>Device Integrity</li> <li>Decemp Configuration</li> </ul>	Voltage Alarm For VCC PSEP		SYS RESET	
XilSEM Library Configuration	Voltage Alarm For VCC PMC or VCCALIX PMC		NONE SYS INTERBUPT	
Tamper	Voltage Alarm For VCC_FILE OF VCCADAC_ INC		SVS INTERRIPT	
	Voltage Alam For Vocast	0		
	Voltage Alarm Por VociNi Or VociAbx or Voci_KAM			
	Voltage Alarm For VCCO_503		STS INTERROPT	
	Glitch Detector		SYS INTERRUPT	
	🗹 Tamper Trigger Register		SYS INTERRUPT	
	External from MIO		SYS INTERRUPT	

#### Figure 13: Tamper Events/Response Configuration

### **PS-NoC Interfaces**

The PS-NoC Interfaces tab enables memory-mapped connectivity from the Control, Interfaces, and Processing System processors to other Versal<sup>™</sup> device resources such as DDR, AI Engine, and PL.

The following table lists the NoC interfaces from which you can select in the customization core:



#### Table 1: List of NoC Interfaces

Interface Name	Size	Notes
4 NoC Master Ports	128-bit channels	PS-CCI → NoC channels
2 NoC Master Ports	128-bit channels	$PS-NCI \rightarrow NoC$ channels
1 NoC Master Ports	128-bit channels	PS-LPD (RPU) $\rightarrow$ NoC channels
1 NoC Master Ports	128-bit channels	PMC/Debug → NoC channels(via LPD)
2 NoC Master Ports	128-bit channels	CPM4 (PCIe/CCIX) $\rightarrow$ NoC channels (via LPD)
4 NoC Slave Ports	128-bit channels	NoC $\rightarrow$ PS channels (2 go to CCI, 2 go to NCI)
1 NoC Slave Ports	128-bit channels	NoC $\rightarrow$ PS-LPD/PMC channels
1 NoC Slave Ports	128-bit channels	NoC $\rightarrow$ PCIe/CCIX/CPM4 channels (via LPD)

The CIPS IP core for the NoC interface selection is shown below:

#### Figure 14: PS-NoC Interfaces

ocumentation 🛛 🕞 IP Location		
omponent Name versal_cips_0		
configuration Options	PS Master Interfaces	
Board		
Home		
Boot Mode	PS to NoC	
PS-PMC		
IO Configuration	Non Coherent Interfaces	
Clock Configuration	PS to NoC Interface 0	
PL-PS Interfaces		
NoC	PS to NoC Interface 1	
Interrupts	RPU to NoC	
CPM4		
Device Integrity	PMC to NoC	
	PS Slave Interfaces	
	Coherent Interfaces	
	NoC to PS Interface 0	
	NoC to PS Interface 1	
	Non Cobarant Interfaces	
	☐ NoC to PS Interface 0	
	NoC to PS Interface 1	
	NoC to PMC	

The following are a few recommendations on the usage of PS-NoC ports.

• If any design has AI Engine then you must enable PMC NoC port for AI Engine configuration.



 If any design has AI Engine and connected to RPU over LPD NOC port, then upper 1 GB of DDR low0 region (0x40000000 to 0x7FFFFFF) is allocated to AI Engine from all LPD masters perspective. If any LPD masters like LPDMA, GEM/USB DMA wants to access DDR then these masters transactions should go through FPD. So you should enable route through FPD bit for respective masters.

*Note*: Following is the Tcl command to enable the route through FPD support for LPD Masters:

```
set_property -dict [list CONFIG.PMC_SD0_ROUTE_THROUGH_FPD {1}
CONFIG.PMC_SD1_ROUTE_THROUGH_FPD {1}
CONFIG.PMC_OSPI_ROUTE_THROUGH_FPD {1}
CONFIG.PMC_QSPI_ROUTE_THROUGH_FPD {1}
CONFIG.PS_USB_ROUTE_THROUGH_FPD {1}
CONFIG.PS_GEM0_ROUTE_THROUGH_FPD {1}
CONFIG.PS_GEM1_ROUTE_THROUGH_FPD {1}
CONFIG.PS_LPDMA0_ROUTE_THROUGH_FPD
                                    {1}
CONFIG.PS_LPDMA1_ROUTE_THROUGH_FPD
                                    {1}
CONFIG.PS_LPDMA2_ROUTE_THROUGH_FPD
                                    {1}
CONFIG.PS_LPDMA3_ROUTE_THROUGH_FPD {1}
CONFIG.PS_LPDMA4_ROUTE_THROUGH_FPD {1}
CONFIG.PS_LPDMA5_ROUTE_THROUGH_FPD {1}
CONFIG.PS_LPDMA6_ROUTE_THROUGH_FPD {1}
CONFIG.PS_LPDMA7_ROUTE_THROUGH_FPD {1}]
[get_bd_cells versal_cips_0]
```

• All 4 PS to NoC CCI ports must connect to the NoC.

### **NoC Interfaces**

The following is the list of interfaces between PS/PMC and NoC. You can select these in the NoC interfaces section to communicate with the DDR/PL/AIE.

 Coherent (CCI) master ports: The CIPS IP core has four coherent master ports (FPD\_CCI\_NOC\_0,1,2,3) connected from PS-CCI to NoC. CCI drives these ports in interleaving mode (2 ports and 4 ports), so you must connect all 4 ports to NoC to access any slave. The CIPS core masters A72/R5/PMC/DMA can make use of these ports. Also, PL masters that are connected to CIPS on CCI slave ports can access these ports.

**Note:** Below are the parameters you need to set to assign DDR/PL address regions (which are connected to CIPS master ports) to PL masters which are connected to CIPS slave ports.

```
set_param bd.enableVirtualAddressing 1
set_param bd.enableVirtualAddressing.DDR 1
```

- Non-Coherent (NCI) master ports: The CIPS core has two non-Coherent master ports (FPD\_AXI\_NOC\_0,1). Only PL masters which are connected to NCI slave ports of the CIPS core can access these ports.
- LPD (RPU) master port: There is one master port from LPD (NOC\_LPD\_AXI\_0) to NoC. LPD masters RPU/DMA can make use of this master port to access slaves.
- **PMC master port:** The CIPS core has one master port (PMC\_NOC\_AXI\_0) from PMC domain to NoC. This port is used by PMC for debug/boot.



- **CPM4 master ports:** Two master ports (IF\_PS\_NOC\_PCIE\_0,1) are exposed from the CIPS core. One is connected to PCIeO controller and the other is to CCIX module. CPM4 can access DDR/PL/AIE regions using these ports. You can select connected ports in the CPM4 Configuration page.
- **Coherent (CCI) Slave ports:** The CIPS core has two coherent slave ports (NOC\_FPD\_CCI\_0,1). Masters connected to these ports can achieve coherency and virtualization. Masters connected to these ports can access DDR, PL slaves which are connected to CIPS via CCI ports. Also PL masters have access to CIPS internal memory regions.
- Non-Coherent (NCI) Slave ports: The CIPS core has two non-coherent slave ports (NOC\_FPD\_AXI\_0,1). Masters connected to these ports can achieve only virtualization. PL masters connected to these ports can access DDR, PL slaves which are connected to CIPS via NCI ports. Also PL masters have access to CIPS internal memory regions.
- **PMC/LPD Slave port:** There is one slave port (NOC\_PMC\_AXI\_0) to LPD/PMC region. PL masters connected to this port will get access to these regions.
- **CPM4 Slave port:** The CIPS core has one NoC slave port (IF\_NOC\_PS\_PCIE\_0) connected to CPM4 module. External masters can connect to this port to configure the CPM4. You can select this port in the CPM4 Configuration page.

Figure 14: PS-NoC Interfaces shows different NoC master/slave port options to enable these ports.

The following table shows the addresses you can assign to DDR/AI Engine/PL slaves which are connected to CIPS master NoC ports. For more information on NoC address ranges and configuration, see *Versal ACAP Programmable Network on Chip and Integrated Memory Controller LogiCORE IP Product Guide* (PG313).

Slave	Region	Start Address	Size
	Low0	0x0	2GB
	Low1	0x80000000	32GB
	Low2	0×C00000000	256GB
DDR	Low3	0x1000000000	734GB
	CH1	0×5000000000	1TB
	CH2	0x6000000000	1TB
	CH3	0x7000000000	1TB
DI	PLNOC2TB	0x20100000000	2044GB
ΓL.	PLNOC8TB	0×8000000000	8TB
AI Engine	AIE_0	0x2000000000	4TB

#### Table 2: NoC Region Address

The following figures describe CIPS + DDR + PL slave connections on NoC:



#### Figure 15: CIPS NoC



#### Figure 16: NoC General Configuration

Show disabled ports	Component Name axi_noc_0		
	General Inputs Outputs Com	nectivity QoS DDR Basic DDR Memory DDI	R Address Mapping DDR Advance
	AXI Interfaces		
+ S00_AXI + S01_AXI + S02_AXI + S02_AXI	Number of AXI Slave Interfaces 6 Number of AXI Master Interfaces 1 Number of AXI Clocks 6		
+ 504 AXI	Inter-NoC Interfaces		
+ \$05_AXI + \$y\$_clk0 + \$y\$_clk0 + \$y\$_clk1 CH0_DDR4_0 + + \$y\$_clk1 CH0_DDR4_1 + + \$y\$_clk2 CH0_DDR4_2 +	Number of Inter-NOC Slave Interface Number of Inter-NOC Master Interface Memory Controllers	es 0 ~ ~ ces 0 ~ ~	
+ sys_clk3 CH0 DDR4 3 +	Memory Controller	x4 Interleaved Memory Controllers	~
- aclk0	Number of Memory Controller Ports	1	~
	Interleave Size in Bytes	128	~
acika	DDR Address Region 0	DDR LOW0 (0x0000 0000 0000 to 0x0000 7FFF FFFF; 20	;) 🗸
- aclk4	DDR Address Region 1	NONE	~
– aclk5	The DDR Address Regions define the For more information on DDR Addres	e possible address locations of DDR Memory Controllers. ss Regions, please refer the System Address Map section	in <u>PG313</u>





#### Figure 17: NoC Slave Ports Configuration



#### AXI NoC (1.0)

w disabled ports	Component Name	Component Name awi_noc_0								
	General Input	s Outputs Conne	ctivity QoS	DDR Basic	DDR Memory	DDR Addres	ss Mapping	DDR Advanced		
	AXI Outputs									
+ S00_AXI	+ - +	4 1 C								
+ S01_AXI	Name	Connected 1	0	Clock	Parity	1	Interru	pt		
+ S02_AXI	N00_001	PL.	×	acik0	~	0		0		
+ 503 AXI										
+ 504 AXI										
L 605 AVI										
+ S05_AXI										
+ S05_AXI + S06_AXI										
+ \$05_AXI + \$06_AXI + \$07_AXI										
+ \$05_AXI + \$06_AXI + \$07_AXI + \$95_CIK0 M00_AXI +	Inter-NoC Outp	outs								
+ S05_AXI + S06_AXI + S07_AXI + sys_clk0 M00_AXI + aclk0 CH0_DDR4_0 +	Inter-NoC Outp	euts								
+ S05_AXI + S06_AXI + S07_AXI + sys_clk0 M00_AXI + aclk0 CH0_DDR4_0 + aclk1	Inter-NoC Outp	euts								
+ \$05_AX1 + \$06_AX1 + \$07_AX1 + \$y\$_clk0 M00_AX1 + aclk0 CH0_DDR4_0 + aclk1 aclk2	Inter-NoC Outp	euts								
+ \$05_AXI + \$06_AXI + \$07_AXI + \$y\$_clk0 M00_AXI + aclk0 CH0_DDR4_0 + aclk1 aclk2 aclk3	inter-NoC Outs	euts ↓ \$ 0 \$								
+ \$05_AXI + \$06_AXI + \$07_AXI + \$ys_clk0 M00_AXI + aclk0 CH0_DDR4_0 + aclk1 aclk2 aclk3 aclk4	Inter-NoC Outp	outs		No con	tent					
+ S05_AXI + S06_AXI + S07_AXI + sys_Clk0 M00_AXI + aclk0 CH0_DDR4_0 + aclk1 aclk1 aclk3 aclk4 aclk5	Inter-NoC Outp	outs ↓ ♣ <mark>0 ¢</mark>		No con	d end					
+ \$05_AXI + \$06_AXI + \$07_AXI + \$ys_Clk0 M00_AXI + aclk0 CH0_DDR4_0 + aclk1 aclk2 aclk4 aclk4 aclk5 aclk6	Inter-NoC Outp	uts ∔ 0 ‡		No con	lard					
+ \$05_AXI + \$06_AXI + \$07_AXI = \$07_AXI = aclk0 CH0_DDR4_0 +    = aclk1 = aclk2 = aclk3 = aclk4 = aclk5 = aclk6 = aclk7	Inter-NoC Outg	uts # 0 \$		No con	tert.					
+ \$05_AXI + \$06_AXI + \$07_AXI - aclk0 M00_AXI + aclk0 CH0_DDR4_0 + - aclk1 - aclk2 - aclk3 - aclk4 - aclk5 - aclk6 - aclk7 - aclk8	Inter-NoC Outp	uts ₿		No con	lert.					

4



how disabled ports	Component	Name ax	i_noc_0						
	General	Inputs	Outputs	Connectivity	QoS	DDR Basic	DDR Memory	DDR Address Mapping	DDR Advance
	Connec	All	isconnect All						
	<u>S</u> earch:	Q٠							
		MOO AX	MC Port 0						
	S00_AXI								
	S01_AXI								
:: + S03_AXI	S02_AXI								
: + S04_AXI	\$03_AXI								
+ S05_AXI	S04_AXI								
+ sys_clk0 CH0_DDR4_0 +    + sys_clk1 CH0_DDR4_1 +    + sys_clk2 CH0_DDR4_2 +    + sys_clk3 CH0_DDR4_3 +    - aclk1 - aclk1 - aclk2 - aclk3		J	0						
- acik4									
– aclk5									

#### Figure 19: NoC Connectivity Configuration

# **Clocking Configuration**

This page enables you to configure the peripheral clocks, PL clocks, DDR, AIE, and CPU clocks.

### Clocking

There are three clock groups as follows.

- Main Clock Group (MCG). This group has the following PLLs:
  - 。 RPU PLL
  - 。 APU PLL
  - PMC PLL
  - NoC PLL
  - 。 CPM4 PLL
- RTC Clock Group (RCG). This is a real-time clock, and a dedicated internal clock for RTC. A clock divider is not required for this clock.



• Interface Clock Group (ICG). This group has clocks that are provided externally, like the clocks from the physical-side interface (PHY) and PL.

Note: PL side peripherals can be operated through a PL clock (PL\_REF\_CLK).

### Input Clocks

Following are the two input clocks.

- Input Reference frequency : This is the frequency of the clock that is coming from the onboard clock source. There can be two reference clocks: REF\_CLK and PL\_ALT\_REF\_CLK.
- **Peripheral Reference frequency:** This section lists the clock pins and the input frequencies for the peripherals where the clock is driven by MIO pins.

control, interfaces & Processin	g 5ystem (2.17								
Documentation 📑 IP Location									
Component Name versal_cips_0									
Configuration Options	Innut Clocks Output	Clocks			-				
Board Home	← Q								
> Boot Mode	Search: Q-	Search: Q-							
> Debug	Name	Source	Input Freq (MHz)	Range (MHz)					
IO Configuration	<ul> <li>Input Reference freque</li> </ul>	incy							
Clock Configuration	REF_CLK	REF_CLK	33.333 6	27.00 : 60.00					
PL-PS Interfaces	PL_ALT_REF_CLK	PL_ALT_REF_CLK	33.333 0	27.00 : 60.00					
NoC	PMC IRO CLK	PMC IRO CLK	400	+/-15%					
Interrupts	Peripheral Reference fr	Peripheral Reference frequency							
> Device Integrity									
					~				
	ć 🔍								

#### Figure 20: Input Clocks Configuration

### **Output Clocks**

This section displays the default/user selected peripheral clocks which are allowed to update the frequency. Also output clocks hold different domain PLLs.

PLLs in the PS and PMC are:

- APLL: APU PLL which is in FPD domain
- NPLL: NoC PLL which is in PMC domain
- RPLL: RPU PLL which is in LPD domain
- PPLL: PMC PLL which is in PMC domain



In the default mode (when the manual mode is turned off), the core automatically chooses the source PLLs and calculates the M (Multiplier) and D (Divisor) values, to ensure that the tool meets the requested frequency to the nearest possible value. The core might not achieve all the requested values, since each PLL caters to multiple peripherals. An internal algorithm creates the best possible solution based on the following conditions.

The algorithm chooses source PLL on its own and the rule is the PMC domain PPLL, NPLL can be used to source in LPD and FPD. The LPD domain RPU PLL can be used to source in and FPD, vice-versa is possible only by setting cross domain PLL parameter.

- When Ethernet is enabled, the core tries to give the precedence to the solution which has the Ethernet frequency of 125 MHz. In manual mode divisors should be made in order to obtain either of 125/25/2.5 MHz.
- When Ethernet is enabled and if there are multiple clocking solutions with the identical Ethernet frequency of 125 MHz, then the tool will take the precedence of the solution that will have the least possible total error (sum of requested frequencies-sum of actual frequencies) value of various peripherals.
- The tool will also take the precedence of the solution with least possible total error value of various peripherals even when the Ethernet is disabled.
- The tool will generate CAN clocks within 0.25 % tolerance and GEM clocks with +/- 100 ppm tolerance. If the tool unable to derive these with set of input clocks then it generates a DRC.
- Tool will generate SDIO ref clock and SD DLL clocks as mentioned below.
  - 1. In auto mode fixed 200 MHz for SDIO0/1 and 1200 MHz for SD DLL.
  - 2. In manual mode DRC is provided if you are not using the same PLL for SD DLL and SDIO0/1 ref clock.
  - 3. In manual mode DRC is provided if SD DLL ref clock is not = 6 times SDIO0/1 ref clock.



ocumentation 📄 IP Location											
omponent Name versal_cips_0											
Configuration Options	Input Clocks	Output Cl	ocke								
Board Home Boot Mode	Manual Cl	ocking Mode	UCKS								
PS-PMC	PLL Name	Source		FBDIV	CLKC	UTDIV	PLL Output (MH	Hz) Cross dom	ain Path	Divisor	Output Freq(MHz)
10 Configuration	APLL	REF CLK	~	81	2	~	1349.9865	APLL_TO_XF	D_CLK	1	1349.9865
Clock Configuration	NPLL	REF CLK	~	120	4	~	999.99	NPLL_TO_XF	PD_CLK	1	999.99
EL-PS Interfaces	RPLL	REF CLK	~	99	2	~	1649.9835	RPLL_TO_XF	D_CLK	2	824.99176
NoC Interrupts	PPLL	REF CLK	~	72	2	~	1199.988	PPLL_TO_XP	PD_CLK	1	1199.988
Device Integrity	← Q ¥ Search: Q Name	\$	So	urce		Request	ed Freq (MHz)	Divisor 0 A	ctual Fre	equency (	MHz) Range
	V PMC Domai	n Clocks	22								
	✓ Processo	or/memory Cloc	KS .					[]			
	M HSN	10	PP	'LL	~	33.333	8	36 33	.333000	)	0.00:200.00
	MSN	11	PP	LL	~	133.333	$\otimes$	9 13	3.33200	01	0.00 : 200.00
	> Peripher	als/10 Clocks									
	> PL Fabric	Clocks									
	> Intercon	nect and Swite	h clock	s							
	V Low Power Domain Clocks										
	> Processor/Memory Clocks										
	> Processo										
	> Processo > Peripher	als/10 Clocks									
	> Processo > Peripher > System D	als/10 Clocks Jebug Clocks									
	<ul> <li>&gt; Processo</li> <li>&gt; Peripher</li> <li>&gt; System D</li> <li>&gt; Intercom</li> </ul>	als/IO Clocks lebug Clocks nect and Switc	h clock	s							
	<ul> <li>&gt; Processo</li> <li>&gt; Peripher</li> <li>&gt; System D</li> <li>&gt; Intercom</li> <li>&gt; Full Power D</li> </ul>	als/10 Clocks Debug Clocks Dect and Switc Domain Clocks	h clock	s							

#### Figure 21: Output Clocks

#### **Enable Manual Clocking Mode**

When you select this mode, different options are displayed. You can directly input the Source PLL, M and D values for various PLLs as well as individual peripheral clock divisor values enabling finer control. In Manual clocking mode, the default divisor values are given for input Ref clock frequency of 33.33 MHz. If you move to the manual mode with different ref clock frequency, then you will encounter DRC's for divisor values which user need to resolve manually.

#### **PMC Power Domain Clocks**

- **Processor/Memory Clocks :** Clock configuration for the HSM0, which is source for AIE PLL. HSM1 which is source for DDR PLL.
- **Peripherals/IO Clocks :** Clock configuration for boot devices like OSPI, SD/eMMC and clocks for NPI, NoC.
- Interconnect and switch clocks : Clock configuration for interconnects and switches in PMC domain.

#### **Low Power Domain Clocks**

- **Processor/Memory Clocks :** Clock configuration for the CPU\_R5 Processor.
- Peripherals/IO Clocks : Clock configuration for low-speed peripheral devices.



- Interconnect and switch clocks : Clock configuration for interconnects and switches in LPD domain.
- System Debug Clocks : Clock configuration for debug modules DBG\_LPD, DBG\_TSTMP.

#### **Full Power Domain Clocks**

- **Processor/Memory Clocks :** Clock configuration for APU, GPU, and DDR
- System Debug Clocks : Clock configuration for debug modules: DBG\_FPD
- Interconnect and Switch clocks : Clock configuration for interconnects and switches in FPD domain

#### PL Clocks

The Versal Control, Interfaces, and Processing System provides four clocks to the PL. Versal CIPS IP core enables the configuration of these clocks to be used in the PL. The Versal CIPS core inserts a BUFG for each of the PL clocks. Also, PCW provide option to select IRO clock to enable and connect to PL peripherals.

You can use PMC domain PLL's in FPD and LPD but the reverse is not allowed because only forward path clocking is followed in CIPS.

Output Clock	Description
Source	This is the source PLL for the corresponding peripheral
Requested Freq (MHz)	This is the input frequency given to the corresponding peripheral
Divisor 0	Denotes the 6-bit programmable Divisor
Actual Freq (MHz)	This is the actual frequency calculated by the Processor Configuration. The clocking algorithm works with multiple factors, peripherals, PLLs, and priorities. Therefore, in certain cases, the actual frequency might be different than the requested frequency.
Range (MHz)	This is the minimum/maximum range of the frequency that the corresponding peripheral can work with. In this mode, you must configure the M and D values to achieve the desired frequency. When this mode is enabled, the values requested through the output mode will be overwritten.

#### Table 3: Output Clocks and their Descriptions

Note: In order to modify the clock frequencies/divisors, the corresponding clock must be enabled.

#### **PLL Options for Output Clocks**

There are four PLLs available in the Versal<sup>™</sup> PS and PMC that are spread across the 3 domains, PMC, LPD and FPD. There are two PLLs namely PPLL and NPLL in the PMC domain while the RPLL in the LPD domain and APLL in FPD domain. The Control, Interfaces, and Processing System IP core provides an option to make use of the cross domain PLLs to be used to source the cross-over peripheral. This gives additional options to select from a pool of all PLLs.



#### Table 4: PLL Options

PLL Option	Description
Name	One of the four PLLs available in APLL, RPLL, PPLL, and NPLL.
Source	This is the source PLL for the corresponding peripheral.
Multiplier (FBDIV)	Denotes the 6-bit Integer value which will be used as multiplier in calculating the respective PLL output frequency.
CLKOUTDIV	Enable the divide by 2/4/8 function inside the PLL. The output of this will be the actual output frequency of respective PLL.
PLL output (MHz)	Final output frequency of the respective PLL.
Cross domain Paths	Denotes the cross-domain name as APLL_TO_LPD for FPD PLLs, NPLL/ PPLL_TO_FPD for PMC PLL's and RPLL_TO_FPD for LPD PLLs.
Divisors	Denotes the 6-bit integer value. This value will be used as divisor in calculating the cross-domain output frequency for respective PLL.
Output frequency (MHz)	PLL output frequency for cross domain.

In the Auto mode, you may not get the actual frequency that is requested due to the load of different clocks on the same source PLL. After instantiating the CIPS IP core, some clocks are enabled by default as per the clocking sheet of the respective part.

# I/O Configuration

This page enables peripherals and their IO connectivity. You can assign attributes for the signals. The I/O peripherals are categorized into two domains PMC and PS. There are total 78 MIOs, 52 in PMC region (PMC MIO Bank0 and PMC MIO Bank1) and 26 in PS region (LPD MIO Bank). Each IO can be assigned to any peripheral based on rules.

Alternatively, the same pins from each peripheral can be routed to EMIO signals which brings the signal to PL section of the device for further processing. For more information on the MIO and EMIO, refer to the Multiplexed I/O in the *Versal ACAP Technical Reference Manual* (AM011). MIOs available for peripheral pinouts are divided into three Banks: PMC MIO Bank0 (MIO 0-25), PMC MIO Bank1 (MIO 26-51), and LPD MIO Bank (MIO 52-77). Each bank has a common I/O Voltage Standard for all its MIOs and the default value for this is LVCMOS1.8 and there are two more options of LVCMOS2.5 and LVCMOS3.3 I/O voltage standard.

You can select the peripherals in core to make use of the MIOs. DRC messages will be shown to alert if any MIO conflict occurs between multiple peripherals. Each peripheral has different set of supported MIO where you can play between these to avoid the DRC of MIO conflict between peripherals. Also, you have EMIO option for each peripheral, this option also can be selected to resolve MIO conflicts.

Upon enabling the peripheral in IO configuration page, you will able set the respective peripheral frequency in Clock Configuration.



For each MIO, there are set of pin pad attributes where user can set these attributes in the core by clicking on respective MIO.

- **Drive:** MIO pin pad attribute Drive Strength in mA, used to select the drive strength. Possible values are 2, 4, 8, and 12.
- Slew: MIO pin pad attribute Speed, specifies whether the device is fast or slow depending on the slew rate.
- **Pull:** MIO pin pad attribute Pull Type, used to enable/disable a device along with pull up or pull down.
- Schmitt: MIO pin pad attribute I/O type, select CMOS or Schmitt as the input I/O voltage type.
- **Direction:** MIO pin pad attribute Direction, the direction can be fixed for certain signals.

The following diagram shows the PS/PMC MIO banks and MIO pad attributes settings.

Documentation 📄 IP Location											
omponent Name versal_cips_i	0										
Configuration Options	MIO Voltage Standard	1									
Board	Bank0 (MIO 0:251 Ba	nk1_IMI0_26:511	Bank2 (MIO 0:)	251 Ban	k3 (Dedicated)						
Home											
Boot Mode	LVCMOS1.8 V LVCMOS1.8 V LVCMOS1.8 V										
Debug											
PS-PMC	Q ≟ ≑ ≔										
10 Configuration	Search: Q-				PMC Bank 0		PMC Bank 1		PS Bank 2		
Clock Configuration	-	110		MIO 0	QSPI: qspi0_clk	MI0 26		MIO 0			
PL-PS Interfaces	Peripheral PMC-Domain	1/0	^	MIO 1	QSPI: qspi0_io[1]	MIO 27		MIO 1			
NoC		lan en		MIO 2		MI0 28		MIO 2			
CPMA	V 🕑 QSPI	Single V	~	MIO 3	OSBI: gooi0 io[0]	MIO 29		MIO 3			
> CPM4 > Device Integrity	QSPI Data Mode	×1	~	MIO 5	OSPI: gspi0_iotoj	MI0 30		MIO 4			
	Clock	PMC MID 6	~	MIO 6	dou n dobuo_co_p	MI0 32		MIC 6			
			_	MIO 7		MI0 33		MIO 7			
	OSPI	Single	~	MIO 8		MI0 34	4	MIO 8			
	~ 🗆 SD0	PMC MI0 13 2!	~	MIO 9		MI0 35		MIO 9			
	Plat West	CD 2.0		MIO 10		MI0 36		MIO 10			
	Side Type	30 2.0	~	MI0 11 MI0 12		MIO 37 MIO 38 MIO 39		MIO 11			
	~ 🗆 SD1	PMC MIO 0 11	~					MI0 12			
	Slot Type	SD 2.0	~	MIO 13				MIO 13			
		PMC MID 2 3	~	MI0 14	MI0 14 MI0 40		MI0 15				
		1110 1110 2 11 0		MIO 16		MI0 42		MIO 16			
	External Tamper Ev	NONE	~	MI0 17		MI0 43		MI0 17			
	SMAP	32 Bit	~	MIO 18		MIO 44		MI0 18			
	✓ PMC GPI0			MI0 19		MIO 45		MI0 19			
		0		MI0 20		MIO 46		MI0 20			
		v		MIO 21		MI0 47		MIO 21			
	✓ PS-Domain			MI0 22		MIO 48		MI0 22			
	CAN-FD0	PMC MI0 8 9	$\sim$	MI0 23		MI0 49		MI0 23			
	C CAN ED1	PS MI0 16 17	~ ~	MI0 24		MIU 50		MI0 24			

#### Figure 22: IO Configuration



### MIO I/O Reservation

The MIO Reservation feature allows you to select the unused/unassigned MIO's as GPIO/AUX-IO. To select these, you are required to click the **MIO PIN view** button and then select **GPIO/ AUX-IO** options in the External Usage column for the respected MIO. If any MIO is allocated to a peripheral then that MIO cannot be set as GPIO/AUX-IO, so its External Usage drop down is disabled.

If the MIO usage is GPIO, then you can set its output data as active-High or active-Low and direction as In or Out. After boot, this value will be driven on IO when it is set in Out direction. If the MIO usage is AUX-IO, only then you can set direction as In/Out.

Documentation 🛸 IP Location	1															
omponent Name versal_cips_	0															
Configuration Options	MIQ Voltar	e Standa	ərd													
Board	Bank0 (Mi	0 0:25]	Bank1 (MIO 26:51)	Bank2 (MIO 0:25)	Bank3	(Ded	licated)									
Home > Boot Mode	LVCMOS1.	8 ~	LVCMOS1.8 🗸	LVCMOS1.8 🗸	LVCMO	S1.8	~									
V PS-PMC	Q ₹ ≑	Ξ														
IO Configuration	Search: Q-															
PL-PS Interfaces	MIO Pin	Periphe	eral Signal	Exte	rnal Usagi	е	Output Da	ata	Speed		Pull Type		Schmitt	Drive Strength(mA)	Direction	i
NoC	PMC MIO 0	QSPI	qspi0_clk	Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	out	~
Interrupts > CPM4	PMC MIO 1	QSPI	qspi0_io[1]	Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	in	~
> Device Integrity	PMC MIO 2			Una	ssigned	~	default	~	slow	v	pullup	~		8mA 🗸	inout	~
	PMC MIO 3			GPIO			default	$\sim$	slow	v	pullup	~		8mA v	inout	~
	PMC MIO 4	QSPI	qspi0_io(0)	AUXIO	) signed		default	$\mathbf{v}$	slow	~	pullup	~	2	8mA 🗸	out	~
	PMC MIO 5	QSPI	qspi0_cs_b	PL	loighed		default	Ŷ	slow	v	pullup	~	~	8mA 🗸	out	Ŷ
	PMC MIO 6			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	in	~
	PMC MIO 7			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	inout	~
	PMC MIO 8			Una	ssigned	~	default	~	slow	Ŷ	pullup	~		8mA 🗸	inout	~
	PMC MIO 9			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	inout	~
	PMC MIO 10			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	out	~
	PMC MIO 11			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	in	~
	PMC MIO 12			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	in	~
	PMC MIO 13			Una	ssigned	~	default	~	slow	~	pullup	~		8mA 🗸	inout	~
	PMC MIO 14			Una	ssigned	~	default	~	slow	~	pullup	~		8mA v	inout	~
													-	- ·	1	

Figure 2	3: MIO I/	O Reservation	Settings
----------	-----------	---------------	----------

The IO-Configuration page also allows you to select 64 GPIO-EMIO pins in PMC domain and 32 GPIO-EMIO pins in PS domain. Upon enabling this, these pins will be exposed to the PL region.

### **MIO Ports**

In the Versal<sup>™</sup> design tools, Control, Interfaces, and Processing System IP core is used to configure the core Multi-Use IO (MIO) ports. There are up to 78 MIO ports available from the CIPS IP core. This core allows you to choose the different peripheral ports to be connected to the MIO ports.



#### **Extended MIO Ports**

Since there are only up to 78 MIO ports available, many peripheral I/O ports beyond these can still be routed to the programmable logic through the Extended MIO (EMIO) interface. Alternative routing for IOP interfaces through programmable logic enables you to take full advantage of the IOP available in the CIPS IP core. The EMIO for I2C, SPI flash memory, Gigabit Ethernet Management Data Input/Output (MDIO), SD/eMMC, GPIO 3-state enable signals are inverted in the Versal CIPS IP core. The Versal CIPS IP core allows you to select GPIO up to 96 signals. The Versal CIPS IP core has control logic to adjust user-selected width to flow into CIPS IP core.

## **PL-PS Configuration**

The PL-PS Configuration page controls which interfaces are exposed on the block to the PL. Figure 24: PL-PS Interfaces illustrates the PL-PS interfaces in Control, Interfaces, and Processing System IP core, where we have two GP master ports to PL and three GP slave ports from the PL. Adding to that we have one ACE and one ACP ports from the PL. All of them have a maximum data width of 128 bits.

### AXI4 I/O Compliant Interfaces

Following are the AMBA® AXI4 compliant interfaces:

- Two PS General Purpose Master interfaces user configurable as 32, 64, and 128 bits in width. The default width is 128.
- Three PL General Purpose Master interfaces user configurable as 32, 64, and 128 bits in width. The default width is 128.
- A 128-bit PL Master AXI coherency extension (ACE) interface for coherent I/O to CCI module.
- A 128-bit PL Master ACP interface to support L2 cache allocation from PL masters. Limited to 64-byte cache line transfers.





Oocumentation 🛸 IP Location	
omponent Name versal_cips_0	
Configuration Options	PL Resets
Board Home	Number of PL Resets 4 ~
Boot Mode	PL Interfaces
> Debug < PS-PMC	AXI Master Interfaces
IO Configuration	□ M_AXI_FPD Data ₩dth 128 ~
Clock Configuration	
PL-PS Interfaces	
NoC Interrupts	AXI Slave Interfaces
CPM4	□ S AXI FPD Data Width 128 V
> Device Integrity	
	GP2 (AXI4 version of S_ACE_Lite_FPD interface) Data Width 128 ~
	♥ S_AXI_LPD Data Width 128 ✓
	128
	□ S_ACP_PD 64
	S_ACE_FPD 32
	Perinheral Fabric Signals
	Ethernet
	GEMO FIFO Enable GEMO PTP Enable
	GEM TSU Clock Ports
	ADMA

#### *Figure 24:* **PL-PS Interfaces**

The CIPS IP core has two master ports to PL and three slave ports from the PL. Adding to that we have one ACE and one ACP port from the PL. All of them have a maximum default data width of 128 bits and can be selectable as 32-bit, 64-bit or 128-bit.

The following are the details of the PL interfaces:

- ACE: A full ACE slave port (S\_ACE\_FPD) allowing 2-way coherency between the APU and a PL master. The PL masters can also snoop APU caches via APU ACP port.
- ACP: A 1-way coherency slave port (S\_ACP\_FPD) directly connected the APU, allowing external PL master to allocate memory directly into the L2 cache.
- AXI Slave Ports:
  - Two AXI slave ports (S\_AXI\_FPD, S\_AXI\_GP2/S\_CCI\_FPD) allowing PL masters direct access to the PS not via the NoC
    - PL Masters connected to S\_AXI\_FPD have access to the following:
      - Complete PS subsystem
      - DDR, PL slaves which are connected to CIPS NCI port
      - PL slaves which are connected to M\_AXI\_FPD port
    - PL Masters connected to GP2 port have access to the follwing:
      - Complete PS subsystem



- DDR, PL slaves which are connected to CIPS CCI port
- PL slaves which are connected to M\_AXI\_FPD port

The GP2 port can be used as AXI4 port or ACE\_LITE port. In GUI if GP2 port is enabled, then it acts as AXI4 port. If you set GP2 port in GUI and below user parameter in tcl promt then it acts as ACE\_LITE port.

set\_property CONFIG.PS\_USE\_ACE\_LITE 1 [get\_bd\_cells /versal\_cips\_0]

- One direct AXI slave port (S\_AXI\_LPD) allowing PL masters access to LPD independent of FPD power state. PL masters connected to this masters has access to complete LPD subsystem.
- AXI Mater Ports:
  - One AXI master port (M\_AXI\_FPD) allowing PS masters and PL masters (which are connected to CIPS S\_AXI\_FPD, GP2 and CIPS NOC slave ports) access to PL slaves.
  - One direct AXI master port (M\_AXI\_LPD) allowing LPD masters access to PL slaves independent of FPD power state

The Versal CIPS IP core provides 4 resets to the PL. It enables the configuration of these resets to be used in the PL, which are asynchronous to any clock.

Using this page, you can also configure the PS to PL interface signals related to Ethernet (FIFO, PTP and TSU) and LPD\_DMA flow control support.

Masters have to choose different addresses for the connected PL slaves. Based on which AXI port the slaves are connected to CIPS, following table shows the possible addresses for PL slaves.

Interface	Region	Start Address	Size	
M_AXI_LPD	LPD_AFI_FS	0×80000000	512MB	
	FPD_AFI_0	0xA4000000	192MB	
	FPD_AFI_1	0×B0000000	256MB	
	FPD_PL8GB	0×40000000	8GB	
	FPD_PL1TB	0x400000000	1TB	

#### Table 5: AXI Region Addresses

### **Programmable Logic Interrupts**

The Control Interfaces and Processing System IP core provides three PS to PL interrupt interfaces (in turn these has wide number of shared interrupts for each peripheral) and 16 PL to PS interrupts. Also, the CIPS IP core has a list of Processor and Debug interrupts.





The Interrupt Configuration tab is used to enable/disable the interrupts between the CIPS core and the PL.

These are broadly categorized as the following:

- **PS to PL :** We have wide number of shared interrupts from different regions of PS (LPD, FPD, and PMC) to PL masters or slaves. You can enable these interrupts separately for each domain and can connect each peripheral interrupt signal to PL logic.
- **PL to PS :** There are 16 PL to PS interrupts that are supported. These are shared interrupts from PL logic to GICs of Real-time Processing Unit (RPU) and Application Processing Unit (APU).
- **High priority PL to PS cores (Processor):** These are Legacy FIQ/IRQ interrupts for RPU/APU from PL. One IRQ and FIQ per CPU will be routed from PL to GIC.
- Inter Processor Interrupt: The Inter Processor Interrupt Block provides the ability for any processing unit to interrupt another processing unit by performing a register write. There are seven IPI channels (IPI 0 through IPI 6), which can be assigned to APU, RPU, and PL.

cumentation 🛸 IP Location				
omponent Name versal_cips_0				
Configuration Options	PS to PL Interrupts			
Board Home Poot Mode		мс		
Debug	PL to PS Interrupts			
PS-PMC	LPD			
Clock Configuration PL-PS Interfaces		□ IRQ3 □ IRQ4 □ IRQ5 □ IRQ6 □ IRQ7		
NoC	FPD			
Interrupts			015	
CPM4			415	
Device integrity				
	Debug			
	Debug			
	Debug Processor Interrupts			
	Debug Processor Interrupts RPU Interrupt APU Inte	rrupt C RPU Event APU_EVENT		
	Debug Processor Interrupts RPU Interrupt APU Inte	rrupt CRPU Event APU_EVENT		
	Debug Processor Interrupts RPU Interrupt APU Inter Inter Processor Interrupt(IPI) Confi	rrupt RPU Event APU_EVENT	Mactor	
	Debug Processor Interrupts RPU Interrupt APU Inte Inter Processor Interrupt(IPI) Confi IPI IPI PMC	rrupt RPU Event APU_EVENT guration Enable	Master PMC	
	Debug Processor Interrupts RPU Interrupt APU Inte Inter Processor Interrupt(IPI) Confi IPI IPI PMC IPI PMC No BUF	rrupt RPU Event APU_EVENT guration Enable	Master PMC PMC	
	Debug Processor Interrupts RPU Interrupt APU Inter Inter Processor Interrupt(IPI) Confi IPI IPI PMC IPI PMC No BUF IPI PSM	rrupt RPU Event APU_EVENT guration Enable	Master PMC PMC PSM	
	Debug Processor Interrupts RPU Interrupt APU Inter Inter Processor Interrupt(IPI) Confi IPI IPI PMC IPI PMC No BUF IPI PSM IPI 0	guration Enable	Master PMC PMC PSM A72	
	Debug Processor Interrupts RPU Interrupt APU Inter Inter Processor Interrupt(IPI) Confi IPI IPI PMC IPI PMC IPI PMC IPI PMC IPI PMC IPI PMC IPI 0 IPI 1	rrupt RPU Event APU_EVENT guration Enable Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø	Master           PMC           PMC           PSM           A72           A72	
	Debug Processor Interrupts RPU Interrupt APU Inter Inter Processor Interrupt(IPI) Confi IPI IPI PMC IPI PMC IPI PMC IPI PSM IPI 0 IPI 1 IPI 2	rrupt RPU Event APU_EVENT guration  Enable	Master           PMC           PMC           PMC           A72           A72           R5 0           R5 1	
	Debug         Processor Interrupts         RPU Interrupt         RPU Interrupt         Inter Processor Interrupt(IIPI) Confir         IPI         IPI PMC         IPI 0         IPI 1         IPI 2         IPI 3	rrupt RPU Event APU_EVENT guration Enable	Master           PMC           PMC           PSM           A72           A72           R5 0           R5 1           S AXI EPD           C MI CCD	

#### Figure 25: Interrupt Configuration

The interrupts from the Control, Interfaces, and Processing System IP core I/O peripherals (IOP) are routed to the PL. The PL can asynchronously assert up to 20 interrupts to the PS cores like APU/RPU.



- 16 interrupt signals are mapped to the interrupt controller as a peripheral interrupt where each interrupt signal is set to a priority level and mapped to one or both CPUs. To use more than one interrupt signal, use a Concat block in the Vivado IP integrator to automatically size the width of the interrupt vector.
- The remaining four PL interrupt signals are inverted and routed to the nFIQ and nIRQ interrupt directly to the signals to the private peripheral interrupt (PPI) unit of the interrupt controller. There is an nFIQ and nIRQ interrupt for each of two CPUs.

The Interrupt IDs are exported to SW and the same ID can be seen in xparameters.h file or see the Versal ACAP Technical Reference Manual (AM011).

# **Debug Settings**

For information on debugging, see Vivado Design Suite User Guide: Programming and Debugging (UG908).

### **PS-PL Cross Trigger**

The PL to PS Cross Trigger inputs are trigger inputs from PL. You can enable these ports to get the trigger events from hardware and feed to ILA for analyzing/debugging hardware state.

The PS to PL Cross Trigger outputs can be used to set the debug break points in software to halt the hardware. Once the trigger event is given to the hardware, the software accesses the hardware state for debug.





#### Figure 26: PS-PL Cross Trigger Configuration

Control, Interfaces & Processing System (2.1)



### **PS-PL Trace**

You can generate Trace ATB signal to PL. AMBA Trace Bus interrupt is generated from this option, which is connected to CoreSight module. Coresight gets the debug information from different cores in PL and can feed the same to ILA/CIPS core.





Component Name versal_cips_0 Configuration Options Roard	
Configuration Options	
Board	
	PS-PL Trace
Home > Boot Mode	PL to PS Advanced Trace Bus (ATB) Ports
V Debug	PL to PS System Trace Marcrocell (STM) event port
PS-PI	
> PS-PMC	Enable PS Trace Peripheral (note this is also available on the PS-PMC > 10 Configuration page)
> CPM4	Trace I/0 PMC MIO 30 47 🗸
> Device Integrity	Trace Width 32Bit
	Other Arm
	CoreSight Trace ATB Sources Trace ATB Funnel Funnel
	PS-MIO PMC-MIO CoreSight Trace Peripherals PMC-MIO To Trace Pins

#### Figure 27: PS-PL Trace Configuration

### **BSCAN and CAPTURE**

There are four BSCAN Interfaces which are available to connect to any PL Debug module, such as MicroBlaze Debug Module (MDM), Chipscope Debug, etc. The JTAG boundary scan chain is driven from the PMC TAP through all IO in the device. The BSCAN interfaces are connected to PMC Test Access Point (TAP) controller for Debug/boundary scan purpose. CAPTURE ports provide user control and synchronization over when and how the capture register information task is requested. Only the register flip-flop and latch states can be captured.





#### Figure 28: BSCAN and CAPTURE Configuration

Control, Interfaces & Processing System (2.1)



### **HIGH Speed Debug Port (HSDP)**

To meet the debug objectives and to accomplish use cases, Versal™ device has an inbuilt, flexible, and high speed debug subsystem called high speed debug port (HSDP). HSDP is similar to and replacement of ARM-DAP in Versal device. Though HSDP has many sub blocks and distributed into many domains of Versal platform but can be defined for simplicity as a combination of DPC and four different host interfaces. The four host interfaces are available in Versal device as part of HSDP subsystem are:

- JTAG Bridge in PMC
- Hard Aurora in PS
- CPM4 PCIe Controller
- Soft Aurora in PL fabric





Figure 29: HSDP Block Diagram

The overall organization of the HSDP is centered on the DPC. The DPC main data path is through 2 AXI-S ports:

- Ingress AXI-S: A slave AXI-S port where command packets arrive from/through these sources/paths.
  - JTAG PMC\_TAP BSCAN Bridge
  - GT CPM4 XPIPE Aurora Wrapper
  - GT CPM4 PCIe Controller Ingress DMA
  - Soft Aurora in PL Fabric
- Egress AXI-S: A master AXI-S port where reply packets depart through/to these paths/ destinations.
  - BSCAN Bridge ->PMC\_TAP -> JTAG
  - Aurora Wrapper -> CPM4 XPIPE -> GT
  - Egress DMA ->CPM4 PCIe Controller -> GT
  - Soft Aurora in PL Fabric

The major datapath of the HSDP is AXI-S. All other types of signaling JTAG, Gigabit (GT), PCIe, AXI-4, XPIPE etc. are converted into AXI-S by conversion blocks including BSCAN Bridge, Ingress DMA, and Egress DMA. Wherever clock domain crossing is necessary, an AXI-S Async Bridge is employed, and wherever 2 AXI-S buses meet, a mux and demux are used.





#### Figure 30: High Speed Debug Port

### **Boot Mode**

Versal<sup>™</sup> device boots differently from traditional FPGAs. There is no longer a standalone bitstream, but instead in Versal device has a programmable device image (PDI) that includes a PL configuration frame data (CFI). Within the PDI the BootROM and the PLM are responsible for configuring the Versal device.

The PLM includes boot device configuration. Select the boot devices in the Boot Mode page. Multiple boot devices can be selected as supported on the board. Clock settings such as required frequency for boot peripherals and REF\_CLK frequency can be set in this page. If you want to use only the PL section, you should use this page to configure boot peripherals.

QSPI, OSPI, SD0, SD1, eMMC1, and SelectMAP are all primary boot peripheral options. The SD0, SD1, and eMMC1 settings provide flags if the MIO selected is not supported for primary boot.

STARTUP options are available to interface device pins and logic to the global asynchronous set/ reset signal, the global 3-state dedicated routing, and the end of startup (EOS).

The STARTUP options supported in Tcl command prompt, you can set CONFIG.PS\_USE\_STARTUP to one to get this primitive ports on CIPS.



ocumentation 🕞 IP Location							
mponent Name versal_cips_0							
configuration Options							
Home	Input Reference Clock						
Boot Mode	REF_CLK 33.3334 0 [27.000000 - 60.000000]						
IO Configuration							
Debug	Boot Peripherals						
PS-PMC	QSPI						
CPM4	OSPL OSPL REF CLK (MHz) 300 [0.000000 - 300.000000]						
Device integrity	Astron Econo unione (Attach 1980)						
	Mode Dual Stacked V Actual Frequency (MHz) 300						
	Data Mode x4 🔗						
	Leeeback Clock 10 PMC MID 6 V						
	OSPI						
	C) acros 05PI REE (1K (MH2) 200 (0.000000, 200.000000)						
	OSPI_NEF_CLK (MHZ) 200 [0.000000 - 200.000000]						
	Mode Dual Stacked V Actual Frequency (MHz) 200						
	SD 07 eMMC 0						
	SD 0 / eMMC 0 SD0_REF_CLK (MH2) 200 [0.00 - 200.00]						
	SD 0 / eMMC 0 SD0_REF_CLX (MHz) 200 [0.00 - 200.00] SD0_D DMC N00 13 25 4 Actual Frequency (MHz) 200 [0.00 - 200.00]						
	SD 0 / eMMC 0           SD0         SD0_REF_CLK (MHz)         200         [0.00 + 200.00]           SD0 10         PMC MIO 13 25         Actual Frequency (MHz)         200						
	SD 0 / eMMC 0           SD0         SD0_REF_CLK (MHz)         200         [0.00 - 200.00]           SD0 IN         PMC MI0 13 25         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         V         V         V         V						
	SD 0 / eMMc 0         SD0_REF_CLK (MHz)         200         [0.00 - 200.00]           SD0 10         PMC MI0 13 25         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         V         V         V         V						
	SD 0 / eMMC 0         SD0_REF_CLK (MHc)         200         [0.00 - 200.00]           SD0 IO         PMC MI0 1325         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         SD 1 / eMMC 1						
	SD 0 / eMMC 0         SD0_REF_CLK (MHz)         200         [0.00 - 200.00]           SD0 10         PMC MIO 1325         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         SD 1 / eMMC 1         [0.00 - 200.00]         [0.00 - 200.00]           SD 1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]						
	SD 0 / eMMC 0         SD0_REF_CUX (MHz)         200         [0.00 - 200.00]           SD0 0         PMC Mio 1325         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         SD 1 / eMMC 1         SD 1 / eMMC 1         SD 1 / eMMC 1           SD1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]						
	SD 0 / eMMC 0           SD0         SD0_REF_CLX (MHz)         200         [0.00 - 200.00]           SD0 in         PMC Mi0 13 25 v         Actual Frequency (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLX (MHz)         200         [0.00 - 200.00]         Actual Frequency (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLX (MHz)         200         [0.00 - 200.00]         Actual Frequency (MHz)         200         [0.00 - 200.00]						
	SD 0/ eMMC 0         SD0_REF_CLK (MHz)         200         [0.00 - 200.00]           SD0 IO         PMC MIO 1325         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC         Actual Frequency (MHz)         200         [0.00 - 200.00]           SD1 / PMC MIO 011         Actual Frequency (MHz)         200         [0.00 - 200.00]						
	SD 0 / eMMC 0         SD0_REF_CLK (MHz)         200         [0.00 - 200.00]           SD0 10         PMC MI0 1325         Actual Frequency (MHz)         200         [0.00 - 200.00]           Slot Type         SD 2.0         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 1         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]           SD1 / eMMC 2         SD1_REF_CLK (MHz)         200         [0.00 - 200.00]						
	SD 0 / eMMC 0         SD 0 <td></td>						
	SD 0 / eMMC 0         SD 0       PMC MIO 1325 v         SD 10       PMC MIO 1325 v         SIG Type       SD 2.0 v         SD 1 / eMMC 1         SD 1 / eMMC 10 0 11 v         Slet Type         SD 2.0 v						
	SD 0 / eMMC 0         SD0       SD0_REF_CUX (MHz)       200       [0.00 - 200.00]         SD0 in       PMC Mio 1325 v       Actual Frequency (MHz)       200         Slat Type       SD 2.0 v       Actual Frequency (MHz)       200       [0.00 - 200.00]         SD 1 / eMMC 1       SD1_REF_CLK (MHz)       200       [0.00 - 200.00]         SD1 / eMMC 1       SD1_REF_CLK (MHz)       200       [0.00 - 200.00]         SD1 in       PMC Mio 011 v       Actual Frequency (MHz)       200       [0.00 - 200.00]         Sol ito       PMC Mio 011 v       Actual Frequency (MHz)       200       [0.00 - 200.00]         Select MAP       SMAP       SMAP       SMAP       SMAP						

#### Figure 31: Boot Mode Configuration

**Note:** For more information on the Versal ACAP Technical Reference Manual (AM011) and Versal ACAP System Software Developers Guide (UG1304).

### **Unsupported Features**

The core provides a Vivado<sup>®</sup> IP integrator configuration of the CIPS configured IP and its I/O. Due to the device flexibility, only the most common features, I/O configurations and peripheral settings are configured by this core.





## Appendix A

# Debugging

This appendix includes details about resources available on the Xilinx<sup>®</sup> Support website and debugging tools.

If the IP requires a license key, the key must be verified. The Vivado<sup>®</sup> design tools have several license checkpoints for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with an error. License checkpoints are enforced by the following tools:

- Vivado Synthesis
- Vivado Implementation
- write bitstream (Tcl command)



**IMPORTANT!** IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.

# Finding Help on Xilinx.com

To help in the design and debug process when using the core, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support. The Xilinx Community Forums are also available where members can learn, participate, share, and ask questions about Xilinx solutions.

### Documentation

This product guide is the main document associated with the core. 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<sup>®</sup> 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 be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use 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.

### **Technical Support**

Xilinx provides technical support on the Xilinx Community Forums for this LogiCORE<sup>™</sup> 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 ask questions, navigate to the Xilinx Community Forums.

# **Debug Tools**

There are many tools available to address CIPS design issues. It is important to know which tools are useful for debugging various situations.

### Vivado Design Suite Debug Feature

The Vivado<sup>®</sup> Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx<sup>®</sup> devices.



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 the Vivado Design Suite User Guide: Programming and Debugging (UG908).





# Appendix B

# Additional Resources and Legal Notices

### **Xilinx Resources**

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

## **Documentation Navigator and Design Hubs**

Xilinx<sup>®</sup> Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado<sup>®</sup> IDE, select Help → Documentation and Tutorials.
- On Windows, select Start → All Programs → Xilinx Design Tools → DocNav.
- At the Linux command prompt, enter docnav.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

Note: For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

# References

These documents provide supplemental material useful with this guide:



- 1. Vivado Design Suite User Guide: Designing with IP (UG896)
- 2. Vivado Design Suite User Guide: Logic Simulation (UG900)
- 3. Vivado Design Suite User Guide: Getting Started (UG910)
- 4. Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator (UG994)
- 5. Vivado Design Suite User Guide: Programming and Debugging (UG908)
- 6. Versal ACAP System Software Developers Guide (UG1304)
- 7. Versal ACAP System Monitor Architecture Manual (AM006)
- 8. Versal ACAP Technical Reference Manual (AM011)
- 9. Versal ACAP Register Reference (AM012)
- 10. Versal ACAP CPM CCIX Architecture Manual (AM016)
- 11. Versal ACAP CPM Mode for PCI Express Product Guide (PG346)
- 12. Versal ACAP CPM DMA and Bridge Mode for PCI Express Product Guide (PG347)
- 13. Versal ACAP Programmable Network on Chip and Integrated Memory Controller LogiCORE IP Product Guide (PG313)

## **Revision History**

The following table shows the revision history for this document.

Section	Revision Summary							
12/04/2020 Version 2.1								
Initial release.	Initial release. N/A							

## **Please Read: Important Legal Notices**

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 Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at https:// www.xilinx.com/legal.htm#tos; 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 such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at https:// www.xilinx.com/legal.htm#tos.

This document contains preliminary information and is subject to change without notice. Information provided herein relates to products and/or services not yet available for sale, and provided solely for information purposes and are not intended, or to be construed, as an offer for sale or an attempted commercialization of the products and/or services referred to herein.

#### AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

#### Copyright

© Copyright 2020 Xilinx, Inc. Xilinx, the Xilinx logo, Alveo, Artix, Kintex, Spartan, Versal, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. PCI, PCIe, and PCI Express are trademarks of PCI-SIG and used under license. All other trademarks are the property of their respective owners.

