IP CORE MANUAL

RFSoC Syncbus Interface IP

px_rfsoc_syncbus_intrfc

PENTEK

Pentek, Inc.
One Park Way
Upper Saddle River, NJ 07458
(201) 818–5900
http://www.pentek.com/

Manual Part Number: 807.48427 Rev: 1.1 – November 1, 2018

Manual Revision History

Date	Version	Comments
10/19/18	1.0	Initial Release
11/1/18	1.1	Corrected Sect 1.1, Table 2–2, Table 4–2, Table 4–3, and Table 4–4.

Legal Notices

Copyright

Trademarks

Printed in the United States of America.

Table of Contents

Page

Table of Contents

Page

Chapter 4: Register Space

Table of Contents

Page

Chapter 5: Designing with the Core

Chapter 6: Design Flow Steps

Table of Contents

Page

This page is intentionally blank

IP Facts

Description

Pentek's Navigator™ RFSoC Syncbus Interface Core provides an interface between the front panel Sync Bus signals (gate, sync, PPS and trigger signals) and a Xilinx Ultrascale+ RFSOC. It allows selection of the timing signals required to generate output timing event data. It also allows users to add delays to the input signals to compensate for board skew, cable delays and to aid in board–to–board synchronization when using multiple boards.

This core complies with the ARM® AMBA® *AXI4 Specification* and also provides a control/status register interface. This manual defines the hardware interface, software interface, and parameterization options for the RFSoC Syncbus Interface Core.

Features

- Takes timing signals from the Sync Bus Connector, generates timing event data streams, and transfers them via an AXI4–Stream Interface to the Xilinx Ultrascale+ RFSoC.
- Accepts gate, sync, PPS and trigger signals from the user design
- Generates interrupts for rising and falling edges of selected gate, sync, PPS and trigger signals
- Register access through AXI4–Lite CSR interface for control and status
- Includes an LED drive to indicate the status of the selected source

Table 1–1: IP Facts Table

Core Specifics	
Supported Design Family[a]	Kintex® Ultrascale
Supported User Interfaces	AXI4–Lite and AXI4–Stream
Resources	See Table 2–1
Provided with the Core	
Design Files	VHDL
Example Design	Not Provided
Test Bench	VHDL
Constraints File	Not Provided[b]
Simulation Model	VHDL
Supported S/W Driver	HAL Software Support
Tested Design Flows	
Design Entry	Vivado® Design Suite 2018.2 or later
Simulation	Vivado VSim
Synthesis	Vivado Synthesis
Support	
Provided by Pentek fpgasupport@pentek.com	

a.For a complete list of supported devices, see the *Vivado Design Suite Release Notes*.
b.Clock constraints can be applied at the top level module of the user design.

This page is intentionally blank

Chapter 1: Overview

1.1 Functional Description

The RFSoC Syncbus Interface Core provides an interface between a Xilinx Ultrascale+ RFSoC and the front panel Sync Bus signals. It has an AXI4−Lite Interface to provide user access to the Register Space within the core.

The core contains multiplexers for the gate, sync, PPS, sysref and trigger signals which select the source as either the front panel (via the SOM inputs), internal registers or the user logic.

For the front panel inputs, the user can optionally apply skew compensation at the input to the core for the gate, sync, trig and sysref signals. The option to apply the delay logic is available in the set−up GUI when generating the core, and the delay values are controlled on−the−fly by registers in the CSR block.

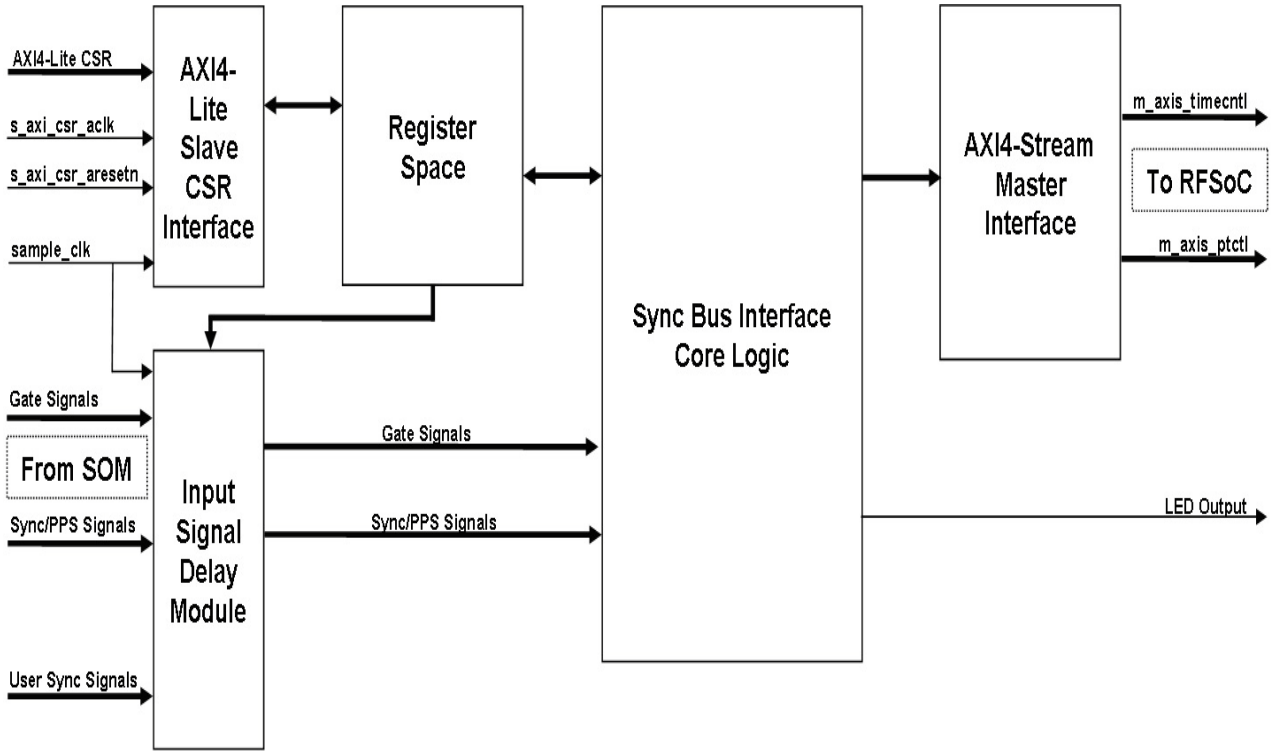
The core also allows the application of a user−configurable "integer" delay to the selected gate and sync signals to allow calibration of multi−board synchronization as well as compensation for long cable delays. The resulting signals are then applied to the outgoing AXI4−Stream.

An AXI4−Lite Control/Status Register (CSR) Bus accesses the control/status registers within the Register Space of the core (see Chapter 4 for a register memory map and bit definitions).

The core also has an LED output with a configurable pulse width expander and a multiplexer to select the driving source.

Figure 1−1 is a top−level block diagram of the Pentek RFSoC Syncbus Interface Core. The modules within the block diagram are explained in the later sections of this manual.

Figure 1–1: RFSoC Syncbus Interface Core Block Diagram



1.1 Functional Description (continued)

- **AXI4–Lite Control/Status Register (CSR) Interface:** This module implements a 32–bit AXI4–Lite CSR Slave Interface to access the control/status and interrupt registers within the Register Space of the core. For additional details about the AXI4–Lite CSR Interface, refer to Section 3.1.

- **Register Space:** This module contains control and status registers including Interrupt Enable, Interrupt Flag, and Interrupt Status registers. Registers are accessed by the user through the AXI4–Lite CSR Interface.

- **Input Signal Delay Module:** This module contains the multiplexer to allow the user to select the desired input source. It then applies the delays designated by the user via the control registers to the selected timing signals. The control registers are accessed by the user through the AXI4–Lite CSR Interface.

- **Sync Bus Interface Core Logic:** This module contains the logic for the interrupts, the LED driver and AXI4–Stream data stream creation. Control registers for these functions are accessed by the user through the AXI4–Lite CSR Interface.

- **AXI4–Stream Master Interface:** This module contains the logic for interfacing to the RFSoC.

1.2 Applications

The Pentek RFSoC Syncbus Interface Core can be incorporated into a Zynq Ultrascale+ RFSoC FPGA to provide access to the module's front panel signals.

1.3 System Requirements

For a list of system requirements, see the Vivado Design Suite Release Notes.

1.4 Licensing and Ordering Information

This core is included with all Pentek Navigator FPGA Design Kits for Pentek Jade series board products. Contact Pentek for licensing and ordering information (www.pentek.com).

1.5 Contacting Technical Support

Technical Support for Pentek's Navigator FPGA Design Kits is available via e–mail (fpgasupport@pentek.com) or by phone (201–818–5900 ext. 238, 9 am to 5 pm EST).

1.6 Documentation

This user manual is the main document for this IP core. The following documents provide supplemental material:

1) *Vivado Design Suite User Guide: Designing with IP*

2) *Vivado Design Suite User Guide: Programming and Debugging*

3) *Xilinx Zynq UltraScale+ RFSoC Data Sheet, DS926*

4) *ARM AMBA AXI4 Protocol Version 2.0 Specification*
http://www.arm.com/products/system-ip/amba-specifications.php

Chapter 2: General Product Specifications

2.1 Standards

The RFSoC Syncbus Interface Core has bus interfaces that comply with the *ARM AMBA AXI4–Lite Protocol Specification* and the *AMBA AXI4–Stream Protocol Specification*.

2.2 Performance

The performance of the RFSoC Syncbus Interface Core is limited by the FPGA logic speed. The values presented in this section should be used as an estimation guideline. Actual performance can vary.

2.2.1 Maximum Frequencies

The RFSoC Syncbus Interface Core has two incoming clock signals. The Sample Clock (**`sample_clk`**) has a maximum frequency of 450MHz and the AXI4–Lite Interface Clock (**`s_axi_csr_aclk`**) has a maximum frequency of 250MHz. Note that 250 MHz is typically the PCIe AXI bus clock frequency.

2.3 Resource Utilization

The resource utilization of the RFSoC Syncbus Interface Core is shown in Table 2–1. Resources have been estimated for a core with a Word Width of 16 (default), Default Packet Size of 256 Bytes and the Packet Length Override option is set to FALSE (default). The target device is a Zynq Ultrascale+ RFSoC XCZU27 –1 speed grade device. These values were generated using the Vivado Design Suite.

Table 2–1: Resource Usage and Availability

Resource	# Used
LUTs	447
Flip–Flops	1,557

NOTE: Actual utilization may vary based on the user design in which the RFSoC Syncbus Interface Core is incorporated.

2.4 Limitations and Unsupported Features

This section is not applicable to this IP core.

2.5 Generic Parameters

The generic parameters of the RFSoC Syncbus Interface Core are described in Table 2–2. These parameters can be set as required by the user application while customizing the core.

Table 2-2: Generic Parameters

Port/Signal Name	Type	Description
has_pin_gate_in	Boolean	**Has Gate In Pin:** Set this parameter to TRUE when a GATE input from the front panel is available. The default setting for this parameter is TRUE.
has_pin_sync_in	Boolean	**Has Sync In Pin:** Set this parameter to TRUE when a SYNC input from the front panel is available. The default setting for this parameter is TRUE.
has_pin_ttl_trig_in	Boolean	**Has TTL Trigger In Pin:** Set this parameter to TRUE when a TTL Trigger input from the front panel is available. The default setting for this parameter is TRUE.
has_pin_sysref_in	Boolean	**Has Sysref In Pin:** Set this parameter to TRUE when a SYSREF input from the front panel is available. The default setting for this parameter is TRUE.
has_pps_in	Boolean	**Has PPS In:** Set this parameter to TRUE when a PPS input from the front panel is available. The default setting for this parameter is TRUE.
has_rtc_pps_in	Boolean	**Has RTC PPS In:** Set this parameter to TRUE when a PPS input from an RTC source is available. The default setting for this parameter is TRUE.
use_gate_tap_delay	Boolean	**Use Gate Tap Delay:** Set this parameter to TRUE when a skew compensation delay is to be applied to the front panel GATE input. The default setting for this parameter is TRUE.
use_sync_tap_delay	Boolean	**Use Sync Tap Delay:** Set this parameter to TRUE when a skew compensation delay is to be applied to the front panel SYNC input. The default setting for this parameter is TRUE.
use_ttl_trig_tap_delay	Boolean	**Use TTL Trig Tap Delay:** Set this parameter to TRUE when a skew compensation delay is to be applied to the front panel TTL Trigger input. The default setting for this parameter is TRUE.

Table 2-2: Generic Parameters (Continued)		
Port/Signal Name	**Type**	**Description**
use_sysref_tap _delay	Boolean	**Use Sysref Tap Delay:** Set this parameter to TRUE when a skew compensation delay is to be applied to the front panel SYSREF input. The default setting for this parameter is TRUE.
initial_gate_tap _delay	Integer	**Initial Gate Tap Delay:** This parameter defines the initial setting for the GATE tap delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**gate_tap_delay**" bits in the Gate Receive Buffer Control register (see Section 4.1) Allowable range is 0 – 511, the default is 0.
initial_sync_tap _delay	Integer	**Initial Sync Tap Delay:** This parameter defines the initial setting for the SYNC tap delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**sync_tap_delay**" bits in the Sync Receive Buffer Control register (see Section 4.2) Allowable range is 0 – 511, the default is 0.
initial_ttl_trig_tap _delay	Integer	**Initial TTL Trigger Tap Delay:** This parameter defines the initial setting for the TTL Trigger tap delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**ttl_trig_tap_delay**" bits in the Gate Receive Buffer Control register (see Section 4.1) Allowable range is 0 – 511, the default is 0.
initial_sysref_tap _delay	Integer	**Initial Sysref Tap Delay:** This parameter defines the initial setting for the SYSREF tap delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**sysref_tap_delay**" bits in the Aux Receive Buffer Control register (see Section 4.3) Allowable range is 0 – 511, the default is 0.
initial_gate_int _delay	Integer	**Initial Gate Integer Delay:** This parameter defines the initial setting for the GATE integer delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**gate_int_delay**" bits in the Gate Receive Buffer Control register (see Section 4.1) Allowable range is 0 – 3, the default is 0.
initial_sync_int _delay	Integer	**Initial Sync Integer Delay:** This parameter defines the initial setting for the SYNC integer delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**sync_int_delay**" bits in the Sync Receive Buffer Control register (see Section 4.2) Allowable range is 0 – 3, the default is 0.
initial_ttl_trig_int _delay	Integer	**Initial TTL Trigger Integer Delay:** This parameter defines the initial setting for the TTL Trigger integer delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**ttl_trig_int_delay**" bits in the Gate Receive Buffer Control register (see Section 4.1) Allowable range is 0 – 3, the default is 0.

Table 2-2: Generic Parameters (Continued)		
Port/Signal Name	**Type**	**Description**
initial_sysref_int_delay	Integer	**Initial Sysref Integer Delay:** This parameter defines the initial setting for the SYSREF integer delay. This setting will be applied to the delay logic upon power–up, but can be overwritten by the user by setting the "**`sysref_int_delay`**" bits in the Aux Receive Buffer Control register (see Section 4.3) Allowable range is 0 – 3, the default is 0.
refclk_freq_int	Integer	**Idelay RefClk Frequency:** This parameter defines the frequency (in MHz) of the reference clock that is applied to the skew delay (Xilinx IDelay) blocks. The default value is 500MHz.
initial_led_src	Integer	**Initial LED Source:** This parameter defines the initial setting for the LED source select. This setting will be applied to the LED select mux upon power–up, but can be overwritten by the user by setting the "**`led_src`**" bits in the Source Select register (see Section 4.4) Allowable range is 0 – 3, the default is 2. The select map is as follows: 0 = Gate 1 = Sync 2 = PPS 3 = Sysref
led_pulse_stretch	Integer	**LED Pulse Stretch:** This parameter defines the setting for the LED Pulse Stretcher. This setting defines the number of **`sample_clk`** cycles during which the LED will be driven following an event on the selected LED source. Allowable range is 0 – 65,535, the default is 65,535.

Chapter 3: Port Descriptions

This chapter provides details about the port descriptions for the following interface types:

- AXI4–Lite Core Interfaces
- AXI4–Stream Core Interface
- I/O Signals

3.1 AXI4–Lite Core Interfaces

The RFSoC Syncbus Interface Core uses the Control/Status Register (CSR) interface to access the control, status and interrupt registers from the user design.

3.1.1 Control/Status Register (CSR) Interface

The CSR interface is an AXI4–Lite Slave Interface that can be used to access the control and status registers in the RFSoC Syncbus Interface Core. Table 3–1 defines the ports in the CSR Interface. See Chapter 4 for the register memory map and bit definitions. See the *AMBA AXI4–Lite Specification* for more details on operation of the AXI4–Lite interfaces.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions

Port	Direction	Width	Description
s_axi_csr_aclk	Input	1	**Clock**
s_axi_csr_aresetn	Input	1	**Reset:** Active low. This value will reset all control/status registers to their initial states.
s_axi_csr_awaddr	Input	6	**Write Address:** Address used for write operations. It must be valid when **s_axi_csr_awvalid** is asserted and must be held until **s_axi_csr_awready** is asserted by the RFSoC Syncbus Interface Core.
s_axi_csr_awprot	Input	3	**Protection:** The RFSoC Syncbus Interface Core ignores these bits.
s_axi_csr_awvalid	Input	1	**Write Address Valid:** This input must be asserted to indicate that a valid write address is available on **s_axi_csr_awaddr**. The RFSoC Syncbus Interface Core asserts **s_axi_csr_awready** when it is ready to accept the address. The **s_axi_csr_awvalid** must remain asserted until the rising clock edge after the assertion of **s_axi_csr_awready**.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions (Continued)			
Port	**Direction**	**Width**	**Description**
s_axi_csr_awready	Output	1	**Write Address Ready:** This output is asserted by the RFSoC Syncbus Interface Core when it is ready to accept the write address.The address is latched when **s_axi_csr_awvalid** and **s_axi_csr_awready** are high on the same cycle.
s_axi_csr_wdata	Input	32	**Write Data:** This data will be written to the address specified by **s_axi_csr_awaddr** when **s_axi_csr_wvalid** and **s_axi_csr_wready** are both asserted. The value must be valid when **s_axi_csr_wvalid** is asserted and held until **s_axi_csr_wready** is also asserted.
s_axi_csr_wstrb	Input	4	**Write Strobes:** This signal, when asserted, indicates the number of bytes of valid data on the **s_axi_csr_wdata** signal. Each of these bits, when asserted, indicate that the corresponding byte of **s_axi_csr_wdata** contains valid data. Bit 0 corresponds to the least significant byte, and bit 3 to the most significant.
s_axi_csr_wvalid	Input	1	**Write Valid:** This signal must be asserted to indicate that the write data is valid for a write operation. The value on **s_axi_csr_wdata** is written into the register at address **s_axi_csr_awaddr** when **s_axi_csr_wready** and **s_axi_csr_wvalid** are high on the same cycle.
s_axi_csr_wready	Output	1	**Write Ready:** This signal is asserted by the RFSoC Syncbus Interface Core when it is ready to accept data. The value on **s_axi_csr_wdata** is written into the register at address **s_axi_csr_awaddr** when **s_axi_csr_wready** and **s_axi_csr_wvalid** are high on the same cycle, assuming that the address has already or simultaneously been submitted.
s_axi_csr_bresp	Output	2	**Write Response:** The RFSoC Syncbus Interface Core indicates success or failure of a write transaction through this signal, which is valid when **s_axi_csr_bvalid** is asserted; 00 = Success of normal access 01 = Success of exclusive access 10 = Slave Error 11 = Decode Error Note: For more details about this signal refer to the *AMBA AXI Specification*.
s_axi_csr_bready	Input	1	**Write Response Ready:** This signal must be asserted by the user logic when it is ready to accept the Write Response.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions (Continued)

Port	Direction	Width	Description
s_axi_csr_bvalid	Output	1	**Write Response Valid:** This signal is asserted by the RFSoC Syncbus Interface Core when the write operation is complete and the Write Response is valid. It is held until **s_axi_csr_bready** is asserted by the user logic.
s_axi_csr_araddr	Input	6	**Read Address:** Address used for read operations. It must be valid when **s_axi_csr_arvalid** is asserted and must be held until **s_axi_csr_arready** is asserted by the RFSoC Syncbus Interface Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the RFSoC Syncbus Interface Core.
s_axi_csr_arvalid	Input	1	**Read Address Valid:** This input must be asserted to indicate that a valid read address is available on **s_axi_csr_araddr**. The core asserts **s_axi_csr_arready** when it ready to accept the Read Address. This input must remain asserted until the rising clock edge after the assertion of **s_axi_csr_arready**.
s_axi_csr_arready	Output	1	**Read Address Ready:** This output is asserted by the RFSoC Syncbus Interface Core when it is ready to accept the read address. The address is latched when **s_axi_csr_arvalid** and **s_axi_csr_arready** are high on the same cycle.
s_axi_csr_rdata	Output	32	**Read Data:** This value is the data read from the address specified by the **s_axi_csr_araddr** when **s_axi_csr_arvalid** and **s_axi_csr_arready** are high on the same cycle.
s_axi_csr_rresp	Output	2	**Read Response:** The RFSoC Syncbus Interface Core indicates success or failure of a read transaction through this signal, which is valid when **s_axi_csr_rvalid** is asserted; 00 = Success of normal access 01 = Success of exclusive access 10 = Slave Error 11 = Decode Error Note: For more details about this signal refer to the *AMBA AXI Specification*.
s_axi_csr_rvalid	Output	1	**Read Data Valid:** This signal is asserted by the RFSoC Syncbus Interface Core when the read is complete and the read data is available on **s_axi_csr_rdata**. It is held until **s_axi_csr_rready** is asserted by the user logic.
s_axi_csr_rready	Input	1	**Read Data Ready:** This signal is asserted by the user logic when it is ready to accept the Read Data.
irq	Output	1	**Interrupt:** This is an active high, edge–type interrupt output representing all of the enabled interrupt sources.

3.2 AXI4–Stream Core Interface

The RFSoC Syncbus Interface Core has the following AXI4–Stream Interface as the output data interface from the core to the Xilinx RFSoC. The signals in this interface are synchronized to **sample_clk**.

3.2.1 AXI4–Stream Master Interface

Table 3–2 defines the ports in the Stream Data Interface. See the *AMBA AXI4–Stream Specification* for more details on the operation of the AXI4–Stream Interface.

.

Table 3-2: AXI4–Stream Master Interface Port Descriptions

Port	Direction	Width	Description
m_axis_timecntl_tdata	Output	32	**Output Data:** This is the output data bus for the time control data. The data mapping is as follows: **tdata[7:0] = Gate** **tdata[15:8] = Sync** **tdata[23:16] = PPS** **tdata[31:24] = Reserved**
m_axis_timecntl_tvalid	Output	1	**Output Data Valid:** This output is asserted by the Pentek RFSoC Syncbus Interface Core when the data on the **m_axis_timecntl_tdata** bus is valid.
m_axis_ptctl_tdata	Output	8	**Output Data:** This is the output data bus for the PTCTL data. The data mapping is as follows: **tdata[0] = Gate** **tdata[1] = Sync** **tdata[2] = PPS** **tdata[31:3] = Reserved**
m_axis_ptctl_tvalid	Output	1	**Output Data Valid:** This output is asserted by the Pentek RFSOC SYNCBUS Interface Core when the data on the **m_axis_ptctl_tdata** bus is valid.

3.3 I/O Signals

The I/O port/signal descriptions of the top level module of the RFSoC Syncbus Interface Core are provided in Table 3–3.

Table 3–3: I/O Signal Descriptions

Port/Signal Name	Type	Direction	Description
sample_clk	std_logic	Input	**Sample Clock:** This clock input provides the clock for all of the logic in the core except for the AXI–Lite CSR interface.
pin_gate_in	std_logic	Input	**Gate Input Pin:** This is the GATE input from the front–panel.
pin_sync_in	std_logic	Input	**Sync Input Pin:** This is the SYNC input from the front–panel.
pin_trig_ttl_in	std_logic	Input	**TTL Trigger Input Pin:** This is the TTL Trigger input from the front–panel.
pps_in	std_logic	Input	**PPS Input Pin:** This is the PPS input from the front–panel.
rtc_pps_in	std_logic	Input	**RTC PPS Input:** This is the PPS input from the RTC module.
pin_sysref_in	std_logic	Input	**Sysref Input:** This is the Sysref input from the front–panel.
pin_trig_ttl_out	std_logic	Output	**TTL Trigger Output:** This is the TTL Trigger output to the front–panel, and is simply the delayed version of the TTL Trigger input.
sysref_tdly_out	std_logic	Output	**Sysref Delayed Output:** This is the Sysref output to the front panel, and is simply the Sysref input with (or without) the skew delay applied.
sysref_out	std_logic	Output	**Sysref Output:** This is the Sysref output. The "**sysref_sel**" bit in the Aux Receive Buffer Control register (see Section 4.3) selects whether this output is the Sysref input with (or without) the skew delay applied, or the clock selected by the "**sysref_rcv_src**" bits in the Source Control Register (see Section 4.4).
user_gate_in	std_logic	Input	**User Gate Input:** This is the GATE input from the user logic.
user_sync_in	std_logic	Input	**User Sync Input:** This is the SYNC input from the user logic.
user_pps_in	std_logic	Input	**User PPS Input:** This is the PPS input from the user logic.
user_sysref_in	std_logic	Input	**User Sysref Input:** This is the Sysref input from the user logic.
led_n	std_logic	Output	**LED Output:** This is the active–LOW output to drive an LED. The source for driving the LED is selected by the "**led_source**" bits in the Source Control Register (see Section 4.4). The LED will be driven for the duration of the time defined by the **led_pulse_stretch** parameter (see Section 2.5).

This page is intentionally blank

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the RFSoC Syncbus Interface Core. The memory map is provided in Table 4–1.

Table 4–1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Gate Receive Buffer Control	0x00	R/W	Controls the delay settings for the Gate and TTL Trigger inputs to the core.
Sync Receive Buffer Control	0x04	R/W	Controls the delay settings for the Sync inputs to the core.
Aux Receive Buffer Control	0x08	R/W	Controls the delay settings for the Sysref inputs to the core.
Source Select	0x0C	R/W	Provides the source selection for the Gate, Sync, PPS, Sysref and LED functions.
Gate Generate	0x10	R/W	This is the user–generated Gate.
Sync Generate	0x14	R/W	This is the user–generated Sync.
PPS Generate	0x18	R/W	This is the user–generated PPS.
Sysref Generate	0x1C	R/W	This is the user–generated Sysref.
Status	0x20	RO	This register provides the current status of selected signals in the core.
Interrupt Enable	0x24	R/W	This register provides enables for the interrupts.
Interrupt Status	0x28	RO	This register provides current status of the interrupts.
Interrupt Flag	0x2C	R/CLR	This register provides status of the interrupt flags.

4.1 Gate Receive Buffer Control Register

This register controls the delay settings for the selected gate and trigger. It is illustrated in Figure 4–1 and described in Table 4–2.

NOTE: The delay element description found below is taken from the Xilinx Zynq UltraScale+ RFSoC Data Sheet, DS926 v1.2.

Figure 4–1: Gate Receive Buffer Control Register

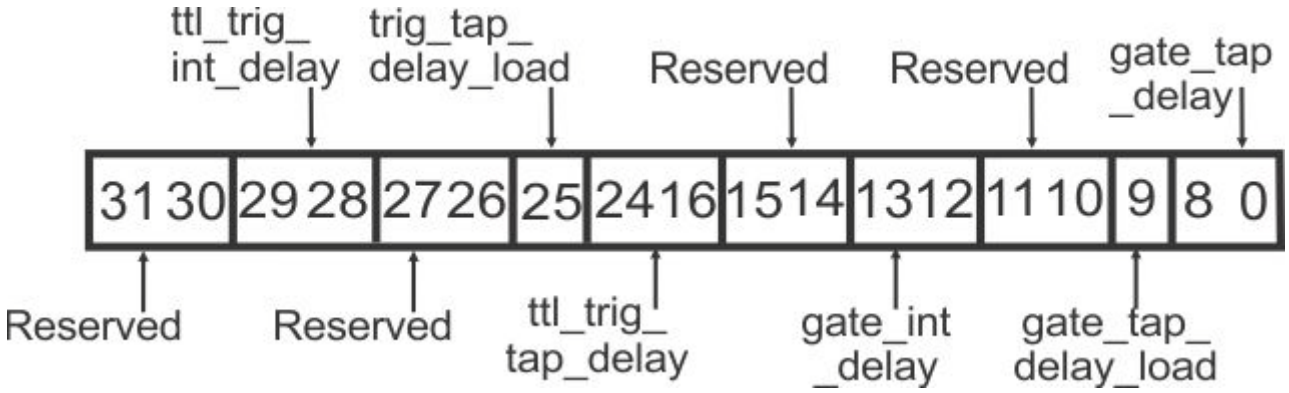


Table 4–2: Gate Receive Buffer Control Register (Base Address + 0x00)

Bits	Field Name	Default Value	Access Type	Description
31:30	Reserved	N/A	N/A	**Reserved**
29:28	ttl_trig_int _delay	initial_ttl _trig_int _delay	R/W	**TTL Trigger Integer Delay:** This value represents the delay that will be added to the selected TTL Trigger source AFTER any (optional) skew delay is applied as follows: "00" = TTL Trigger + skew + 1 x sample_clk "01" = TTL Trigger + skew + 2 x sample_clk "10" = TTL Trigger + skew + 3 x sample_clk "11" = TTL Trigger + skew + 4 x sample_clk
27:26	Reserved	–	–	**Reserved**
25	trig_tap_delay _load	0	R/W	**TTL Trigger Tap Delay Load:** Asserting this signal loads the value in the **`ttl_trig_tap_delay`** bits into the TTL Trigger skew delay module.

24:16	ttl_trig_tap _delay	initial_ttl _trig _tap _delay	R/W	**TTL Trigger Tap Delay:** This value represents the skew delay that will be added to the selected TTL Trigger source. The delay element is a delay chain containing 512 taps, with each tap providing between 2.1ps and 12ps of delay. (See note.) The value in this register determines the number of taps that are applied and, hence, the delay.
15:14	Reserved	–	–	Reserved
13:12	gate_int_delay	initial_ gate_int _delay	R/W	**Gate Integer Delay:** This value represents the delay that will be added to the selected Gate source AFTER any (optional) skew delay is applied as follows: "00" = Gate + skew + 1 x sample_clk "01" = Gate + skew + 2 x sample_clk "10" = Gate + skew + 3 x sample_clk "11" = Gate + skew + 4 x sample_clk
11:10	Reserved	–	–	Reserved
9	gate_tap _delay_load	0	R/W	**Gate Tap Delay Load:** Asserting this signal loads the value in the **`gate_tap_delay`** bits into the Gate skew delay module.
8:0	gate_tap _delay	initial_ gate_ tap_ delay	R/W	**Gate Tap Delay:** This value represents the skew delay that will be added to the selected Gate source. The delay element is a delay chain containing 512 taps, with each tap providing between 2.1ps and 12ps of delay. (See note.) The value in this register determines the number of taps that are applied and, hence, the delay.

4.2 Sync Receive Buffer Control Register

This register controls the delay settings for the selected sync. It is illustrated in Figure 4–2 and described in Table 4–3.

NOTE: The delay element description found below is taken from the Xilinx Zynq UltraScale+ RFSoC Data Sheet, DS926 v1.2.

Figure 4–2: Sync Receive Buffer Control Register

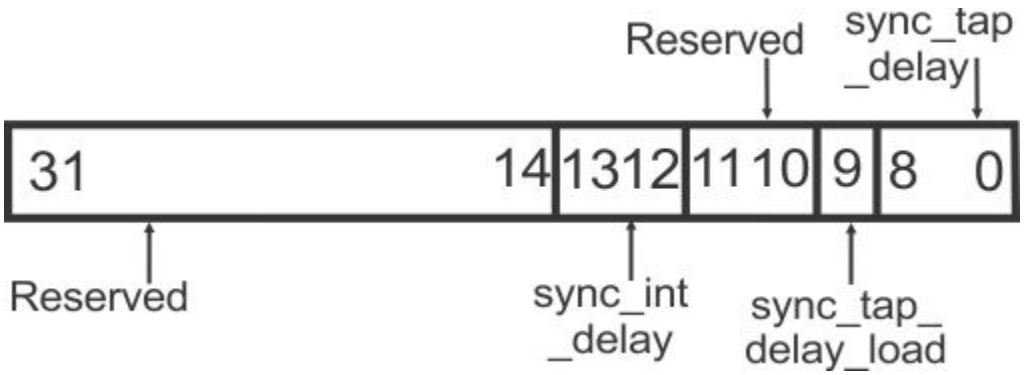


Table 4–3: Sync Receive Buffer Control Register (Base Address + 0x04)

Bits	Field Name	Default Value	Access Type	Description
31:14	Reserved	N/A	N/A	**Reserved**
13:12	sync_int_delay	initial _sync _int _delay	R/W	**Sync Integer Delay:** This value represents the delay that will be added to the selected Sync source AFTER any (optional) skew delay is applied as follows: "00" = Sync + skew + 1 x sample_clk "01" = Sync + skew + 2 x sample_clk "10" = Sync + skew + 3 x sample_clk "11" = Sync + skew + 4 x sample_clk
11:10	Reserved	–	–	**Reserved**
9	sync_tap_ delay_load	0	R/W	**Sync Tap Delay Load:** Asserting this signal loads the value in the **`sync_tap_delay`** bits into the Sync skew delay module.
8:0	sync_tap _delay	initial_ sync_ tap_ delay	R/W	**Sync Tap Delay:** This value represents the skew delay that will be added to the selected Sync source. The delay element is a delay chain containing 512 taps, with each tap providing between 2.1ps and 12ps of delay. (See note.) The value in this register determines the number of taps that are applied and, hence, the delay.

4.3 Aux Receive Buffer Control Register

This register controls the delay settings for the selected sysref. It is illustrated in Figure 4–3 and described in Table 4–4.

NOTE: The delay element description found below is taken from the Xilinx Zynq UltraScale+ RFSoC Data Sheet, DS926 v1.2.

Figure 4–3: Aux Receive Buffer Control Register

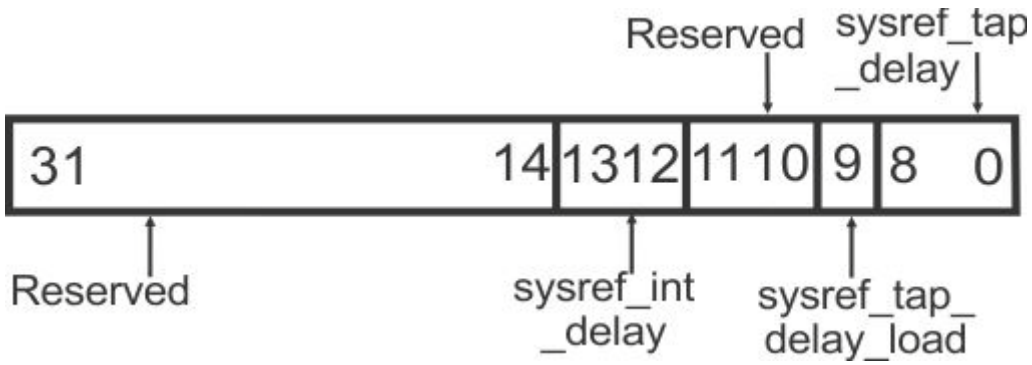


Table 4–4: Aux Receive Buffer Control Register (Base Address + 0x08)

Bits	Field Name	Default Value	Access Type	Description
31:14	Reserved	N/A	N/A	**Reserved**
13:12	sysref_int _delay	initial _sysref _int _delay	R/W	**Sysref Integer Delay:** This value represents the delay that will be added to the selected Sysref source AFTER any (optional) skew delay is applied as follows: "00" = Sysref + skew + 1 x sample_clk "01" = Sysref + skew + 2 x sample_clk "10" = Sysref + skew + 3 x sample_clk "11" = Sysref + skew + 4 x sample_clk
11:10	Reserved	–	–	**Reserved**
9	sysref_tap_ delay_load	0	R/W	**Sysref Tap Delay Load:** Asserting this signal loads the value in the **`sysref_tap_delay`** bits into the Sysref skew delay module.
8:0	sysref_tap _delay	initial_ sysref_ tap_ delay	R/W	**Sysref Tap Delay:** This value represents the skew delay that will be added to the selected Sysref source. The delay element is a delay chain containing 512 taps, with each tap providing between 2.1ps and 12ps of delay. (See note.) The value in this register determines the number of taps that are applied and, hence, the delay.

4.4 Source Select Register

This register controls the input selection multiplexers for the gate, sync, pps, sysref and LED driver logic. It is illustrated in Figure 4–4 and described in Table 4–5.

Figure 4–4: Source Select Register

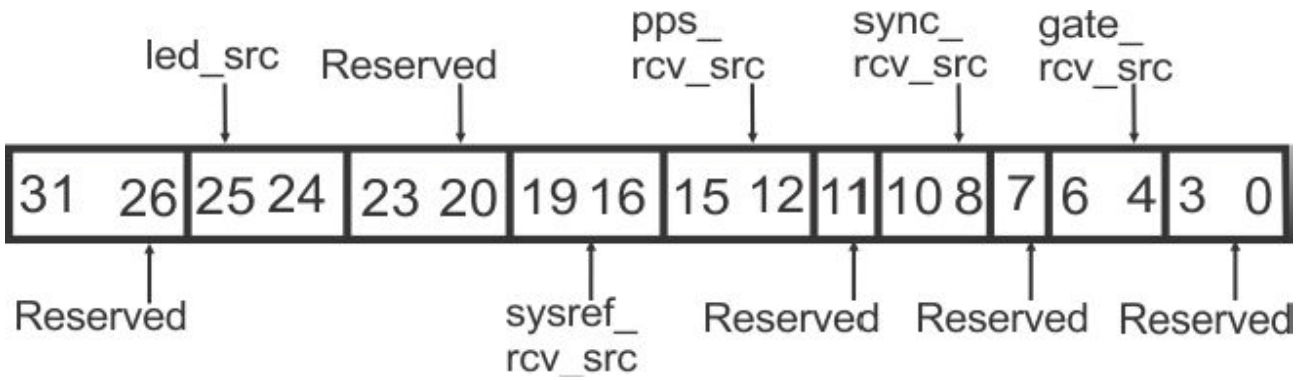


Table 4–5: Source Select Register (Base Address + 0x0C)

Bits	Field Name	Default Value	Access Type	Description
31:26	Reserved	N/A	N/A	**Reserved**
25:24	led_src	initial_ led_src	R/W	**LED Source Select:** These bits select the LED driver input source as follows: "00" = Gate "01" = Sync "10" = PPS "11" = Sysref
23:20	Reserved	–	–	**Reserved**
19:16	sysref_rcv_src	0000	R/W	**Sysref Source Select:** These bits select the sysref source as follows: "0000" = None (off) "0001" = Sysref Register Bit "0010" = Sysref input (Active high) "0011" = Sysref input (Active low) "0100" = Rising Edge of Selected Gate "0101" = Falling Edge of Selected Gate "0110" = Sync Bus Differential Sync (Active high) "0111" = Sync Bus Differential Sync (Active low) "1000" = LVTTL Trig input (Active high) "1001" = LVTTL Trig input (Active low) "1010" = PPS (Active high) "1011" = PPS (Active low) "1100" = User PPS in

15:12	pps_rcv_src	0000	R/W	**PPS Source Select:** These bits select the PPS source as follows: "0000" = None (off) "0001" = PPS Register Bit "0010" = PPS input (Active high) "0011" = PPS input (Active low) "0100" = Rising Edge of Selected Gate "0101" = Falling Edge of Selected Gate "0110" = Sync Bus Differential Sync (Active high) "0111" = Sync Bus Differential Sync (Active low) "1000" = LVTTL Trig input (Active high) "1001" = LVTTL Trig input (Active low) "1010" = RTC PPS (Active high) "1011" = RTC PPS (Active low) "1100" = User PPS in
11	Reserved	–	–	Reserved
10:8	sync_rcv_src	000	R/W	**Sync Source Select:** These bits select the Sync source as follows: "000" = None (off) "001" = Sync Register Bit "010" = Rising Edge of Gate "011" = Falling Edge of Gate "100" = Sync Bus Differential Sync (Active high) "101" = Sync Bus Differential Sync (Active low) "110" = User Sync Input (Active high) "111" = User Sync Input (Active low)
7	Reserved	–	–	Reserved
6:4	gate_rcv_src	000	R/W	**Gate Source Select:** These bits select the Gate source as follows: "000" = None (off) "001" = Gate Register Bit "010" = LVTTL Trig input (Active high) "011" = LVTTL Trig input (Active low) "100" = Sync Bus Differential Gate/Trig (Active high) "101" = Sync Bus Differential Gate/Trig (Active low) "110" = User Gate (Active high) "111" = User Gate (Active low)
3:0	Reserved	–	–	**Reserved**

4.5 Gate Generate Register

The Gate Generate Register is used to manually generate a gate signal. It is illustrated in Figure 4–5 and described in Table 4–6.

Figure 4–5: Gate Generate Register

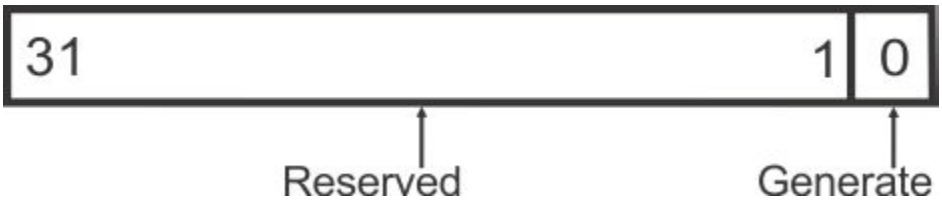


Table 4–6: Gate Generate Register (Base Address + 0x10)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	N/A	N/A	**Reserved**
0	gate_reg_gen	0	R/W	**Gate Generate:** Allows the user to generate a gate signal manually.

4.6 Sync Generate Register

The Sync Generate Register is used to manually generate a sync signal. It is illustrated in Figure 4–6 and described in Table 4–7.

Figure 4–6: Sync Generate Register

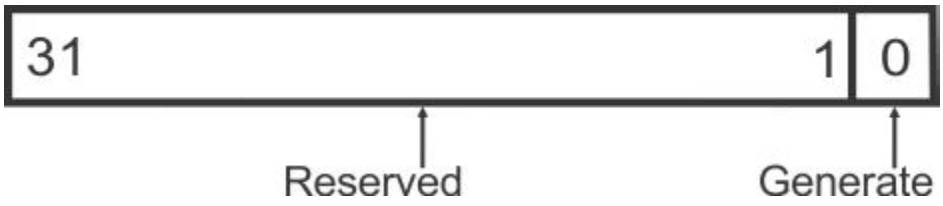


Table 4–7: Sync Generate Register (Base Address + 0x14)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	N/A	N/A	**Reserved**
0	sync_reg_gen	0	R/W	**Sync Generate:** Allows the user to generate a sync signal manually.

4.7 PPS Generate Register

The PPS Generate Register is used to manually generate a PPS signal. It is illustrated in Figure 4–7 and described in Table 4–8.

Figure 4–7: PPS Generate Register

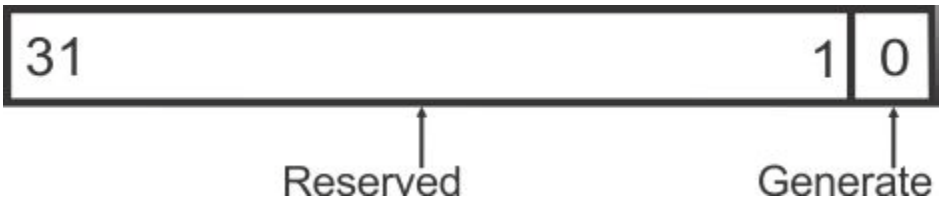


Table 4–8: PPS Generate Register (Base Address + 0x18)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	N/A	N/A	**Reserved**
0	pps_reg_gen	0	R/W	**PPS Generate:** Allows the user to generate a PPS signal manually.

4.8 Sysref Generate Register

The Sysref Generate Register is used to manually generate a Sysref signal. It is illustrated in Figure 4–8 and described in Table 4–9.

Figure 4–8: Sysref Generate Register

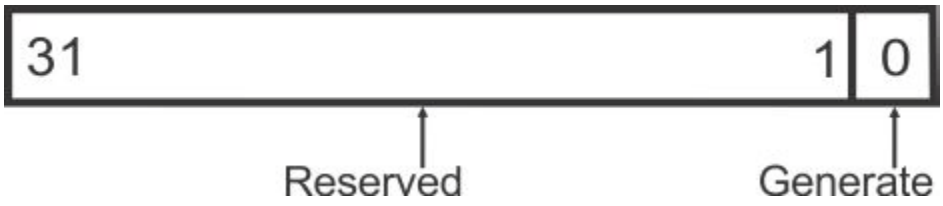


Table 4–9: Sysref Generate Register (Base Address + 0x1C)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	N/A	N/A	**Reserved**
0	sysref_reg _gen	0	R/W	**Sysref Generate:** Allows the user to generate a Sysref signal manually.

4.9 Status Register

This register provides the current states of the selected timing sources. It is illustrated in Figure 4–9 and described in Table 4–10.

Figure 4–9: Status Register

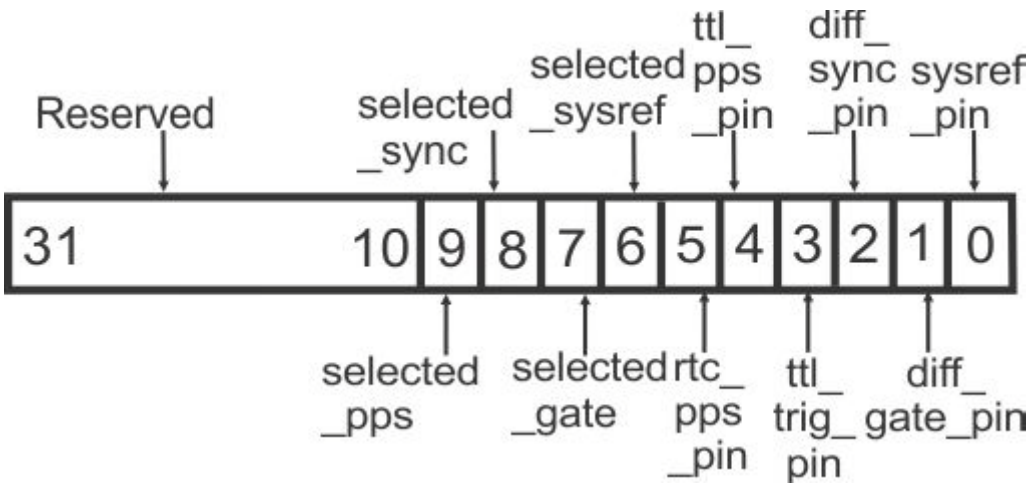


Table 4–10: Status Register (Base Address + 0x20)

Bits	Field Name	Default Value	Access Type	Description
31:10	Reserved	N/A	N/A	**Reserved**
9	selected_pps	0	RO	**Selected PPS:** Provides the current status of the PPS signal that is being applied to the AXI4–Stream Master output bus.
8	selected_sync	0	RO	**Selected Sync:** Provides the current status of the Sync signal that is being applied to the AXI4–Stream Master output bus.
7	selected_gate	0	RO	**Selected Gate:** Provides the current status of the Gate signal that is being applied to the AXI4–Stream Master output bus.
6	selected _sysref	0	RO	**Selected Sysref:** Provides the current status of the Sysref signal that is being applied to the AXI4–Stream Master output bus.
5	rtc_pps_pin	0	RO	**RTC PPS:** Provides the current status of the PPS signal from the RTC module that is coming into the core.
4	ttl_pps_pin	0	RO	**Front Panel PPS:** Provides the current status of the PPS signal from the front panel that is coming into the core.

3	ttl_trig_pin	0	RO	**Post Delay Trigger:** Provides the current status of the trigger signal from the front panel that is coming into the core AFTER any user−configured skew delays.
2	diff_sync_pin	0	RO	**Post Delay Sync:** Provides the current status of the Sync signal from the front panel that is coming into the core AFTER any user−configured skew delays.
1	diff_gate_pin	0	RO	**Post Delay Gate:** Provides the current status of the Gate signal from the front panel that is coming into the core AFTER any user−configured skew delays.
0	sysref_pin	0	RO	**Post Delay Sysref:** Provides the current status of the Sysref signal from the front panel that is coming into the core AFTER any user−configured skew delays.

4.10 Interrupt Enable Register

The bits in the interrupt enable register are used to enable (or disable) the generation of interrupts based on the condition of certain circuit elements, known as interrupt sources. When a bit in this register associated with a given interrupt source is High, an interrupt will be generated by the rising edge of that source's Interrupt Status Register bit (See Section 4.15). It is illustrated in Figure 4–10 and described in Table 4–11.

Figure 4–10: Interrupt Enable Register

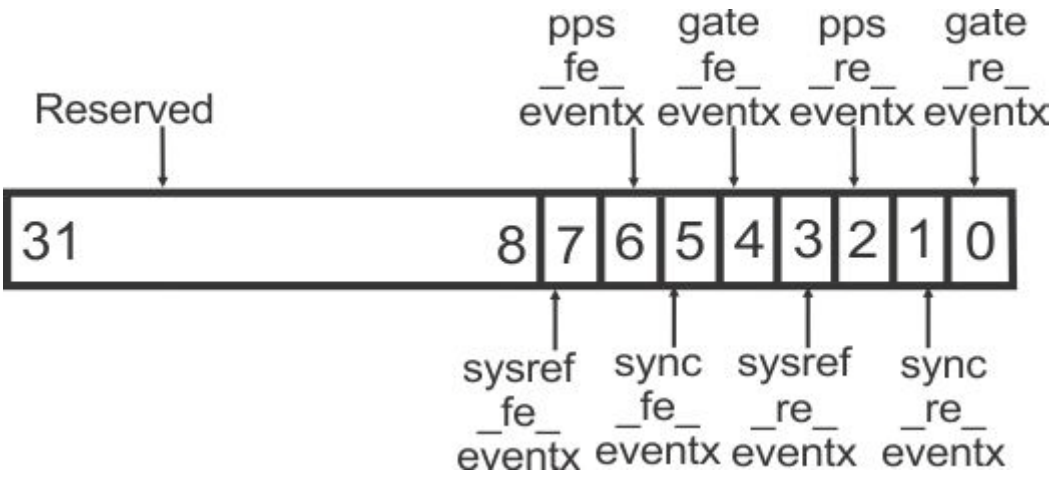


Table 4–11: Interrupt Enable Register (Base Address + 0x24)

Bits	Field Name	Default Value	Access Type	Description
31:8	Reserved	N/A	N/A	**Reserved**
7	sysref_fe _eventx	0	R/W	**Sysref Falling–Edge Event:** This bit enables/disables the Sysref falling–edge event interrupt source. The Sysref falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Sysref signal. '0' = Disable interrupt '1' = Enable interrupt
6	pps_fe_eventx	0	R/W	**PPS Falling–Edge Event:** This bit enables/disables the PPS falling–edge event interrupt source. The PPS falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected PPS signal. '0' = Disable interrupt '1' = Enable interrupt

5	sync_fe _eventx	0	R/W	**Sync Falling–Edge Event:** This bit enables/disables the Sync falling–edge event interrupt source. The Sync falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Sync signal. '0' = Disable interrupt '1' = Enable interrupt
4	gate_fe _eventx	0	R/W	**Gate Falling–Edge Event:** This bit enables/disables the Gate falling–edge event interrupt source. The Gate falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Gate signal. '0' = Disable interrupt '1' = Enable interrupt
3	sysref_re _eventx	0	R/W	**Sysref Rising–Edge Event:** This bit enables/disables the Sysref rising–edge event interrupt source. The Sysref rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Sysref signal. '0' = Disable interrupt '1' = Enable interrupt
2	pps_re_eventx	0	R/W	**PPS Rising–Edge Event:** This bit enables/disables the PPS rising–edge event interrupt source. The PPS rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected PPS signal. '0' = Disable interrupt '1' = Enable interrupt
1	sync_re _eventx	0	R/W	**Sync Rising–Edge Event:** This bit enables/disables the Sync rising–edge event interrupt source. The Sync rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Sync signal. '0' = Disable interrupt '1' = Enable interrupt
0	gate_re _eventx	0	R/W	**Gate Rising–Edge Event:** This bit enables/disables the Gate rising–edge event interrupt source. The Gate rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Gate signal. '0' = Disable interrupt '1' = Enable interrupt

4.11 Interrupt Status Register

The Interrupt Status Register has read–only access associated with each interrupt condition. A status bit changes to '1' when the source interrupt occurs. When a status bit in this register changes to '1' the corresponding flag bit in the Interrupt Flag Register is set to '1'. A status bit in this register clears to '0' when that interrupt condition clears, whereas the associated flag bit in the Interrupt Flag Register remains at logic '1' until it is explicitly cleared by the user.

Some of the interrupt sources are transient and so may not appear in the Interrupt Status Register at the time it is read. In such cases, use the Interrupt Flag Register to see the interrupt conditions that have occurred. It is illustrated in Figure 4–11 and described in Table 4–12.

Figure 4–11: Interrupt Status Register

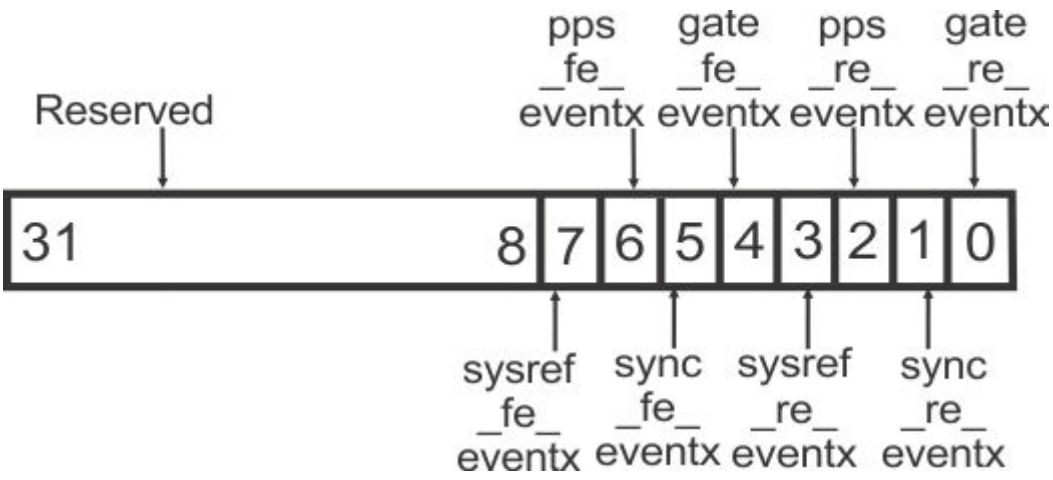


Table 4–12: Interrupt Status Register (Base Address + 0x28)

Bits	Field Name	Default Value	Access Type	Description
31:8	Reserved	N/A	N/A	**Reserved**
7	sysref_fe _eventx	0	RO	**Sysref Falling–Edge Event:** This bit indicates the status of the Sysref falling–edge event interrupt source. The Sysref falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Sysref signal. '0' = No interrupt '1' = Interrupt condition asserted

6	pps_fe_eventx	0	RO	**PPS Falling−Edge Event:** This bit indicates the status of the PPS falling−edge event interrupt source. The PPS falling−edge event interrupt source indicates that there has been a falling−edge transition on the selected PPS signal. '0' = No interrupt '1' = Interrupt condition asserted
5	sync_fe _eventx	0	RO	**Sync Falling−Edge Event:** This bit indicates the status of the Sync falling−edge event interrupt source. The Sync falling−edge event interrupt source indicates that there has been a falling−edge transition on the selected Sync signal. '0' = No interrupt '1' = Interrupt condition asserted
4	gate_fe _eventx	0	RO	**Gate Falling−Edge Event:** This bit indicates the status of the Gate falling−edge event interrupt source. The Gate falling−edge event interrupt source indicates that there has been a falling−edge transition on the selected Gate signal. '0' = No interrupt '1' = Interrupt condition asserted
3	sysref_re _eventx	0	RO	**Sysref Rising−Edge Event:** This bit indicates the status of the Sysref rising−edge event interrupt source. The Sysref rising−edge event interrupt source indicates that there has been a rising−edge transition on the selected Sysref signal. '0' = No interrupt '1' = Interrupt condition asserted
2	pps_re_eventx	0	RO	**PPS Rising−Edge Event:** This bit indicates the status of the PPS rising−edge event interrupt source. The PPS rising−edge event interrupt source indicates that there has been a rising−edge transition on the selected PPS signal. '0' = No interrupt '1' = Interrupt condition asserted
1	sync_re _eventx	0	RO	**Sync Rising−Edge Event:** This bit indicates the status of the Sync rising−edge event interrupt source. The Sync rising−edge event interrupt source indicates that there has been a rising−edge transition on the selected Sync signal. '0' = No interrupt '1' = Interrupt condition asserted
0	gate_re _eventx	0	RO	**Gate Rising−Edge Event:** This bit indicates the status of the Gate rising−edge event interrupt source. The Gate rising−edge event interrupt source indicates that there has been a rising−edge transition on the selected Gate signal. '0' = No interrupt '1' = Interrupt condition asserted

4.12 Interrupt Flag Register

The Interrupt Flag Register has read/clear access associated with each interrupt condition. When reset, this register has all bits set to '0' (cleared). Each flag bit in this register latches an interrupt occurrence. A '1' in any flag bit in this register indicates that an interrupt has occurred.

Note that when any status bit in the Interrupt Status Register, changes to '1' the corresponding flag bit in this register will also be set to '1'. However, when a status bit in the Interrupt Status Register clears from '1' to '0, the corresponding latched flag bit in this register does not clear, but remains at '1'. To clear the flag bits, write '1's to the desired bits. The flags are not affected by the Interrupt Enable Register. It is illustrated in Figure 4–12 and described in Table 4–13.

Figure 4–12: Interrupt Flag Register

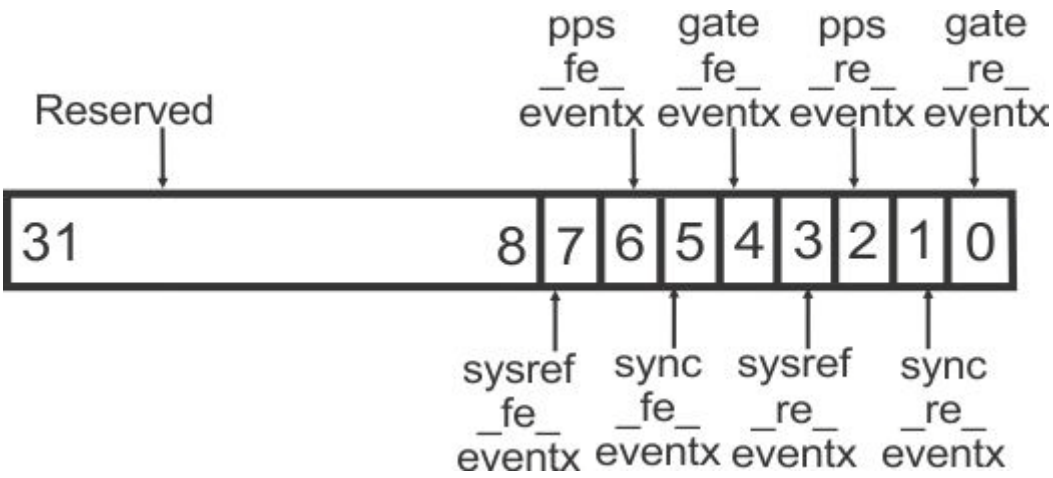


Table 4–13: Interrupt Flag Register (Base Address + 0x2C)

Bits	Field Name	Default Value	Access Type	Description
31:8	Reserved	N/A	N/A	**Reserved**
7	sysref_fe_eventx	0	R/CLR	**Sysref Falling–Edge Event:** This bit indicates the status of the Sysref falling–edge event interrupt flag. The Sysref falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Sysref signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
6	pps_fe_eventx	0	R/CLR	**PPS Falling–Edge Event:** This bit indicates the status of the PPS falling–edge event interrupt flag. The PPS falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected PPS signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch

5	sync_fe _eventx	0	R/CLR	**Sync Falling–Edge Event:** This bit indicates the status of the Sync falling–edge event interrupt flag. The Sync falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Sync signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
4	gate_fe _eventx	0	R/CLR	**Gate Falling–Edge Event:** This bit indicates the status of the Gate falling–edge event interrupt flag. The Gate falling–edge event interrupt source indicates that there has been a falling–edge transition on the selected Gate signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
3	sysref_re _eventx	0	R/CLR	**Sysref Rising–Edge Event:** This bit indicates the status of the Sysref rising–edge event interrupt flag. The Sysref rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Sysref signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
2	pps_re_eventx	0	R/CLR	**PPS Rising–Edge Event:** This bit indicates the status of the PPS rising–edge event interrupt flag. The PPS rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected PPS signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
1	sync_re _eventx	0	R/CLR	**Sync Rising–Edge Event:** This bit indicates the status of the Sync rising–edge event interrupt flag. The Sync rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Sync signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch
0	gate_re _eventx	0	R/CLR	**Gate Rising–Edge Event:** This bit indicates the status of the Gate rising–edge event interrupt flag. The Gate rising–edge event interrupt source indicates that there has been a rising–edge transition on the selected Gate signal. '0' = No interrupt '1' = Interrupt latched **Clear:** '1' = Clear latch

This page is intentionally blank

Chapter 5: Designing with the Core

This chapter provides guidelines and additional information to facilitate designing with the RFSoC Syncbus Interface Core.

5.1 General Design Guidelines

The Pentek RFSoC Syncbus Interface Core provides the required logic to select from multiple timing sources including front panel inputs, user inputs and CSR register–based inputs, and apply the selected timing signals to AXI4–Stream data streams. The user can control the selection of the timing signals as well as add delays to compensate for board skew and cable delays as well as to compensate for board–to–board skew when synchronizing multiple boards by setting the control registers in the Register Space (as described in Chapter 4) to the desired values.

5.2 Clocking

Sample Clock: **`sample_clk`**

This clock is used to clock all of the ports and logic in the Pentek RFSoC Syncbus Interface Core except for the AXI–Lite CSR logic.

CSR Clock: **`s_axi_csr_aclk`**

This clock is used to clock the AXI–Lite CSR logic.

5.3 Resets

Main Reset: **`s_axi_csr_aresetn`**

This is an active low synchronous reset associated with **`s_axi_csr_aclk`**. When asserted, this signal will reset all of the skew delay modules in the core as well as the logic in the CSR block, will return all control registers back to their default settings and will clear the interrupt enables and flags.

5.4 Interrupts

This core has an edge type (rising edge−triggered) interrupt output (**`irq`**), which is synchronous with **`s_axi_csr_aclk`**. On the rising edge of any interrupt signal, a one clock cycle wide pulse is output from the core on the irq output. Each interrupt event is stored in two registers, accessible on the **`s_axi_csr`** bus. The Interrupt Status Register always reflects the current state of the interrupt condition, which may have changed since the generation of the interrupt. The Interrupt Flag Register latches the occurrence of each interrupt, in a bit that retains its state until explicitly cleared. The interrupt flags can be cleared by writing '1' to the associated bit's location. All interrupt sources that are enabled (via the Interrupt Enable Register) are "OR'ed" onto the irq output.

NOTE: All interrupt sources are latched in the Interrupt Flag Register, even when an interrupt source is not enabled (via the Interrupt Enable Register).

NOTE: Because this core uses edge−triggered interrupts, an interrupt condition which remains active after servicing will not cause the generation of a new interrupt. A new interrupt will only be generated by another rising edge on an interrupt source.

5.5 Interface Operation

- **CSR Interface:** This is the control/status register interface, is associated with **`s_axi_csr_aclk`** and is a standard AXI4−Lite Slave interface. Typically, this interface is connected along with other cores' AXI4−Lite interfaces through an AXI4−Lite Crossbar core or a series of AXI4−Lite Crossbar cores that route AXI4−Lite accesses through to the desired core based on the address range.
- **Input Timing Signals Interface:** This interface consists of signals from the front panel and the user design which mark acquisition events that will be formatted into an AXI4−Stream output for use by the Xilinx RFSoC's ARM processor.
- **AXI4−Stream Output Interface:** This is the AXI4−Stream output containing the reformatted timing information to be used by the Xilinx RFSoC's ARM processor.

5.6 Programming Sequence

This section briefly describes the programming sequence for the RFSoC Syncbus Interface Core.

1) Ensure that the Interrupt Flag Register is cleared.

2) Enable the interrupt enable bits based on the user design requirement.

3) Set the control registers with the required values.

4) When done, check the interrupt flag register and clear the interrupts.

5.7 Timing Diagrams

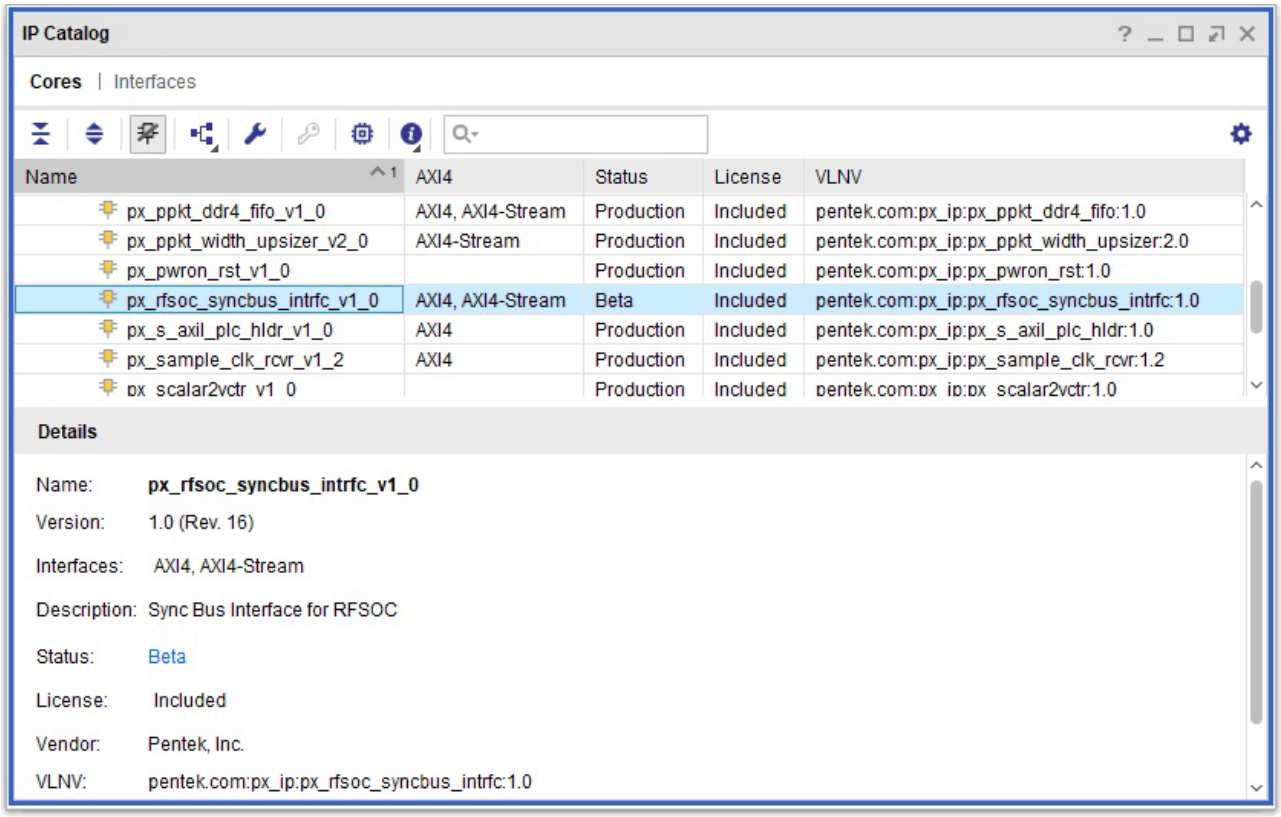
This section is not applicable to this IP core.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek RFSoC Syncbus Interface Core. It also includes simulation, synthesis, and implementation steps that are specific to this IP core. This core can be generated from the Vivado IP Catalog when the Pentek IP Repository has been installed. It will appear in the IP Catalog list as **`px_rfsoc_syncbus_intrfc_v1_0`** as shown in Figure 6–1.

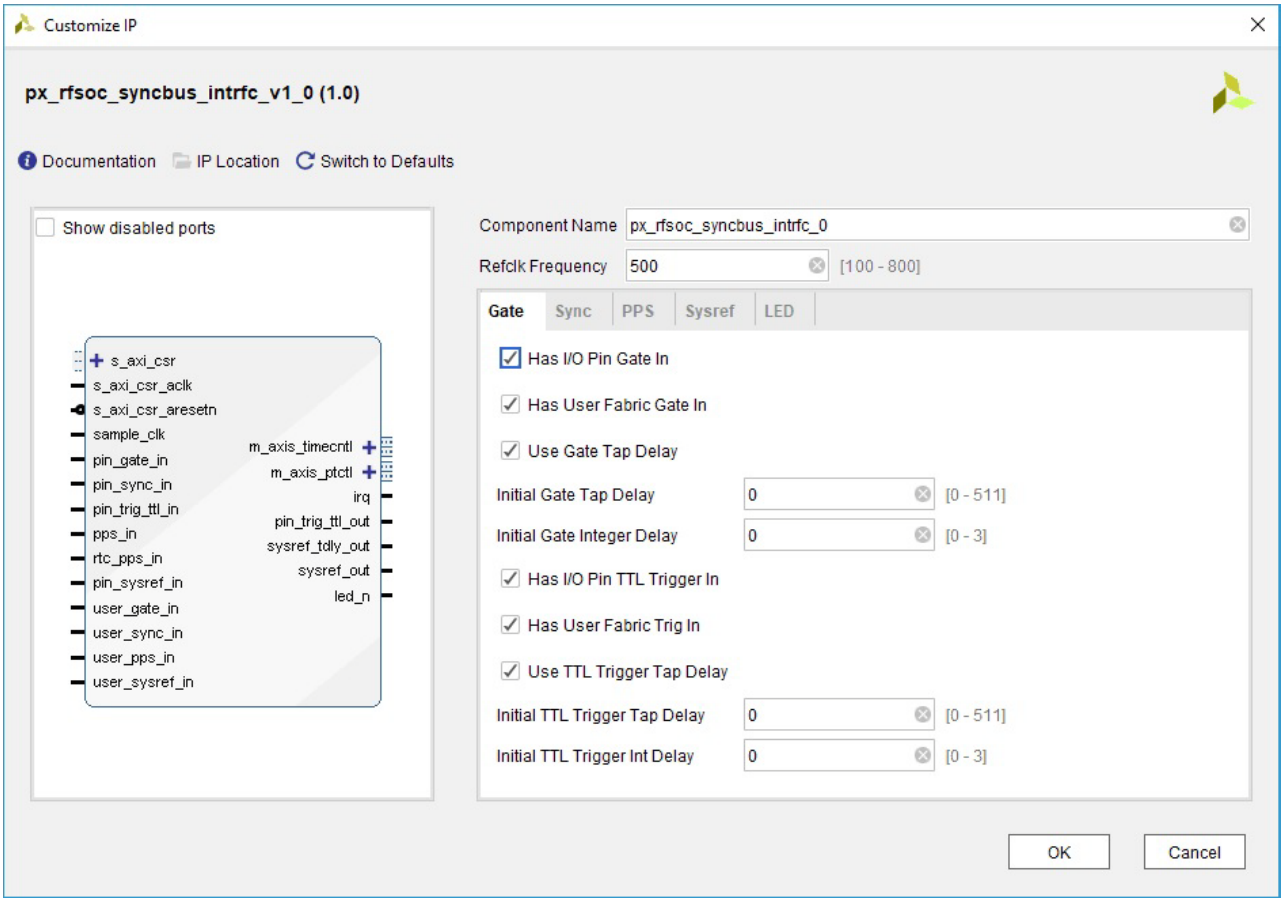
Figure 6–1: RFSoC Syncbus Interface Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **`px_rfsoc_syncbus_v1_0`** core, a screen appears that shows the core's symbol and the core's parameters (see Figure 6–2). The core's symbol is the box on the left side.

Figure 6–2: RFSoC Syncbus Interface Core IP Symbol



6.2 User Parameters

The user parameters of this RFSoC Syncbus Interface Core are explained in Section 2.5 of this user manual.

6.3 Output Generation

For more details about generating and using IP in the Vivado Design Suite, refer to the *Vivado Design Suite User Guide – Designing with IP*.

6.4 Constraining the Core

This section contains information about constraining the RFSoC Syncbus Interface Core in the Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the RFSoC Syncbus Interface Core. The necessary constraints can be applied in the top−level module of the user design.

Device, Package, and Speed Grade Selections

This IP works for the Zynq Ultrascale+ RFSoC FPGAs.

Clock Frequencies

The CSR clock (**`s_axi_csr_aclk`**) of the Pentek RFSoC Syncbus Interface Core can take frequencies up to 250 MHz. The sample clock (**`sample_clk`**) can take frequencies up to 450MHz.

Clock Management

This section is not applicable for this IP core.

Clock Placement

This section is not applicable for this IP core.

Banking and Placement

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

This section is not applicable for this IP core.

6.5 Simulation

This section is not applicable to this IP core.

6.6 Synthesis and Implementation

For details about synthesis and implementation see the *Vivado Design Suite User Guide – Designing with IP*.