IP CORE MANUAL

NAVIGATOR
FPGA Design Kit

8–Channel Channelizer DDC Core

px_8ch_channelizer_ddc

PENTEK

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

Manual Part Number: 807.48392 Rev: 1.0 – June 21, 2017

Manual Revision History

Date	Version	Comments
6/21/17	1.0	Initial Release

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

Table of Contents

Page

Chapter 6: Design Flow Steps

IP Facts

Description

Pentek's Navigator™ 8–channel Channelizer DDC IP Core uses a unique 8–phase radix–8 FFT to pre–tune and decimate the incoming AXI–S data stream, followed by a fine tuning NCO and programmable decimation post filter.

This core complies with the ARM® AMBA® AXI4 specification. This manual defines the hardware interface, software interface, and parameterization options for the 8–channel Channelizer DDC IP Core.

Features

- Programmable Decimation 16 to 1024 (steps of 8)
- Programmable 18–bit Filter Coefficients. Pentek provides default filter coefficient sets for all possible decimations.
- Independently tunable 32–bit NCO tuning frequencies for each channel.
- Independent 32–bit Phase Offset controls for each channel.
- Independently programmable output gain.
- Provides 16–bit or 24–bit output resolutions
- Provides Normal, Inverse, and Offset Spectrum modes.
- Multiple cores can be synchronized
- Efficient use of DSP48E2 Blocks.
- All controls and registers accessible via AXI4–Lite.
- Input and output data streams are AXI4–Stream compatible and configured to interface with Pentek's AXI4–Stream PDTI Type bus definition.
- Multiplexed 8 channel output data stream.
- Independent channel enables
- Gain saturation indication and interrupts.

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	encrypted 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 2017.1 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

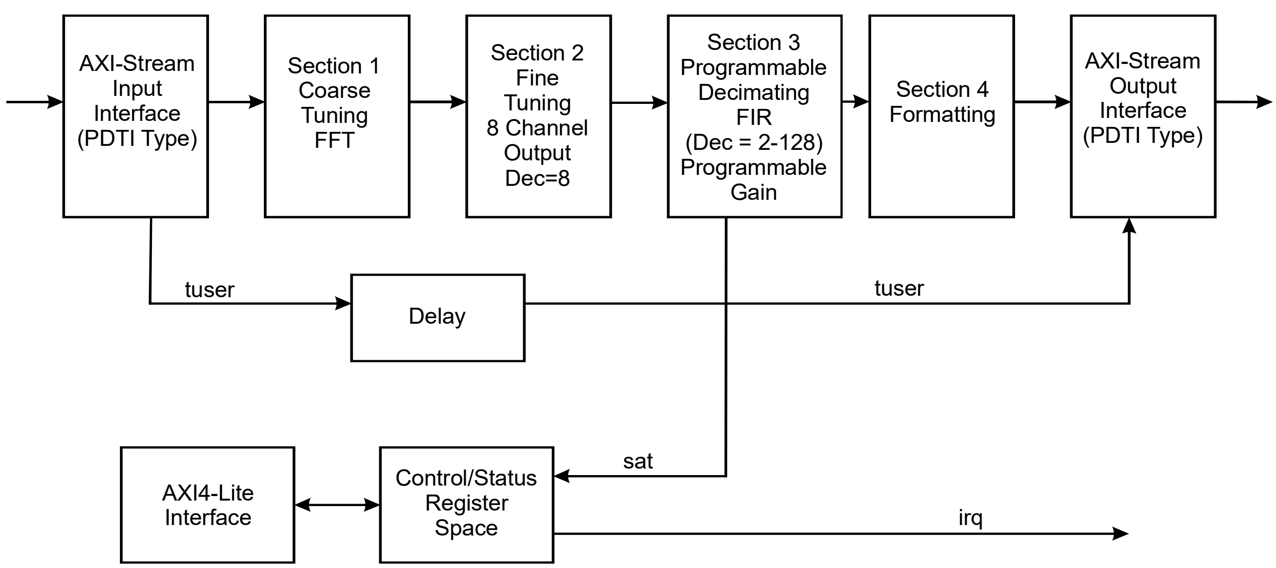
Figure 1–1This core is an 8–channel narrow–band DDC that uses a unique 8–phase radix–8 FFT to pre–tune and decimate the incoming AXI–S data stream, followed by a fine tuning NCO and programmable decimation post filter. This method yields eight independently tunable output channels with a common programmable decimation of 16 through 1024 (steps of 8) with very efficient use of DSP48E2 blocks.

This core accepts a Pentek pdti type AXI–S data stream and outputs a channel multi–plexed pdti type AXI–S data stream.

Timestamp and timing event signals embedded in the Pentek pdti tuser bus are auto–matically delayed through the core to match the filter delay. In other words a time–stamp or timing event signal pulse that occurs coincident to an impulse at the input to the DDC will appear aligned to the impulse response at the output of the DDC. This time delay is calculated and adjusted based on the selected decimation.

Figure 1–1 is the top level block diagram of the 8–channel Channelizer DDC. The mod–ules within the block diagram are explained in the later sections of this manual.

Figure 1–1: 8–channel Channelizer DDC Block Diagram



1.2 Functional Description

- **AXI4–Lite Interface:** This module implements a 32–bit AXI4–Lite Slave interface to access the Register Space. For additional details about the AXI4–Lite Interface, refer to the Section 3.1 AXI4–Lite Core Interfaces.
- **AXI4–Stream Input Interface:** This module implements a 16–bit AXI4–Stream Slave interface (Pentek PDTI type) for the input data stream. For additional details about the AXI4–Stream Interface, refer to the Section 3.2 AXI4–Stream Core Interfaces.
- **Register Space:** This module contains the control and status registers including Interrupt Enable, Interrupt Flag, and Interrupt Status registers. Also included are the coefficient set memory, tuning, synchronization and gain controls. Registers are accessed through the AXI4–Lite interface.
- **Section 1:** This module performs coarse tuning and sample rate decimation via a unique combination of an 8 overlapped phase 64–point FFT and windowing function.
- **Section 2:** This module performs programmable fine tuning for eight output channels.
- **Section 3:** This module performs programmable decimation and output gain adjustment. Decimation can be any integer from 2 to 128, yielding an overall decimation range of 16 to 1024 in steps of 8.
- **Section 4:** This module performs output formatting, controlling output resolution choice of 16 or 24–bit, offset spectrum, and inverse spectrum.

1.3 Applications

Efficient use of DSP48E2 blocks where multiple DDC channels are required from a common input.

1.4 System Requirements

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

1.5 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).

1.6 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).

1.7 Documentation

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

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

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

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

This page is intentionally blank

Chapter 2: General Product Specifications

2.1 Standards

The 8–channel Channelizer DDC core has bus interfaces that comply with the *ARM AMBA AXI4–Lite Protocol Specification* and the *AMBA AXI4–Stream Protocol Specification*. Supports Pentek's AXI4–Stream interface known as PDTI type which includes timestamp, timing events and data type information embedded in the tuser bus.

2.2 Performance

The performance of the DDC Core is limited by the maximum operating frequency of the 350 MHz in a Kintex Ultrascale XCKU060–2 speed grade device. The values presented in this section should be used as an estimation guideline. Actual performance can vary.

2.2.1 Maximum Frequencies

The AXI4–Stream clock (AXIS_ACLK) has a maximum operating frequency of 350 MHz. The AXI4–Lite clock (S_AXI_CSR_ACLK) has a maximum operating frequency of 250 MHz.

2.3 Resource Utilization

The resource utilization of the **px_8ch_channelizer_ddc** is shown in Table 2–1. Resources have been estimated for the Kintex Ultrascale XCKU060 –2 speed grade device. These values were generated using the Vivado Design Suite.

Table 2–1: Resource Usage and Availability

Resource	# Used
LUT	6429
LUTRAM	1682
FF	12119
BRAM	61.50
DSP	103

NOTE: Actual utilization may vary based on the user design in which the px_8ch_channelizer_ddc is incorporated.

2.4 Limitations and Unsupported Features

This core does not support flow control on the AXI4–Stream interfaces. Further, it is mandatory that the input data stream be valid on every clock cycle (tvalid is constantly asserted).

2.5 Generic Parameters

The generic parameters of the **px_8ch_channelizer_ddc** 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
First_chan	Integer	Channel Number of the first of eight channels. This number must be zero or a multiple of eight. Minimum = 0 Maximum = 248

Chapter 3: Port Descriptions

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

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

3.1 AXI4–Lite Core Interfaces

The **px_8ch_channelizer_ddc** uses the Control/Status Register (CSR) interface to con– trol, and receive status from the user design, as well as load the filter coefficient set.

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 DDC Core. Table 3–1 defines the ports in the CSR Interface. See Chapter 4 for a Control/Status 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 registers to their initial states.
s_axi_csr_awaddr	Input	7	**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 ADC Core.
s_axi_csr_awprot	Input	3	**Protection:** The ADC interface core ignores these bits.
s_axi_csr_awvalid	Input	1	**Write Address Valid:** This input must be asserted to indi– cate that a valid write address is available on **s_axi_csr_awaddr**. The ADC 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 ADC 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 **s_axi_csr_wdata** signal. Each of these bits, when asserted indicate that the corre–sponding 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		**Write Ready:** This signal is asserted by the ADC 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 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.
s_axi_csr_bvalid	Output	1	**Write Response Valid:** This signal is asserted by the ADC 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.

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

Port	Direction	Width	Description
s_axi_csr_araddr	Input	7	**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 ADC Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the ADC 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 the **s_axi_csr_araddr**. The ADC 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 ADC 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 ADC 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 ADC Core when the read is complete and the read data is available on the **s_axi_csr_rdata**. It is held until **s_axi_csr_rready** is asserted by the user logic.
s_axi_csr_rready	Input		**Read Data Ready:** This signal is asserted by the user logic when it is ready to accept the Read Data.
irq	Output		**Interrupt:** This is an active High, edge type interrupt request output.

3.2 AXI4–Stream Core Interfaces

The **px_8ch_channelizer_ddc** has the following AXI4–Stream Interface, which is used to transfer data streams.

3.2.1 Input Stream Data (s_axis_pdti) Interface

This interface is used to transfer a 16–bit, single sample per clock cycle PDTI type AXI4–S data stream into the **px_8ch_channelizer_ddc** core. Table 3–2 defines the ports in the Input Stream Data Interface. This interface is an AXI4–Stream Slave Interface. This AXI4–Stream bus is synchronous with the **axis_aclk** clock input of the core. See the *AMBA AXI4–Stream Specification* for more details on the operation of the AXI4–Stream Interface.

Table 3-2: Input Stream Data Interface Port Descriptions

Port	Direction	Width	Description
s_axis_pdti_tdata	Input	16	**Input Data**
s_axis_pdti_tvalid		1	**Output Data Valid:** Asserted when data is valid on the **m_axis_dataio_pd_tdata**. Since ADC data is available every clock cycle, **m_axis_dataio_pd_tvalid** remains High always.
s_axis_pdti_tuser		128	**Input Side–Band data:** The Pentek PDTI definition defines the tuser signals in the following manner: tuser[63:0] = timestamp[63:0] tuser[64] = Gate tuser[72] = Sync tuser[80] = PPS tuser[91:88] = Samples/Cycle (should be 1) tuser[92] = IQ of first sample in data 0 = I, 1 = Q –– 0 (non applicable, set to 0) tuser[94:93] = Data Format (should be 1= 16 bit) tuser[95] = Data Type (should be 0 = Real) tuser[103:96] = channel[7:0] (will be overridden by generic parameter of core) tuser[127:104] = Reserved

3.2 AXI4−Stream Core Interfaces (continued)

3.2.2 Output Stream Data (m_axis_pdti) Interface

This interface is used to output an eight channel multiplexed IQ data stream from the core. Table 3–3 defines the ports in the Input Stream Data Interface. This interface is an AXI4−Stream Master Interface. This AXI4−Stream bus is synchronous with axis_aclk clock input of the core. This interface does not support flow control and has no tready signal. See the *AMBA AXI4−Stream Specification* for more details on the operation of the AXI4−Stream Interface.

3.2 AXI4–Stream Core Interfaces (continued)

Table 3-3: Output Stream Data Interface Port Descriptions			
Port	**Direction**	**Width**	**Description**
m_axis_pdti_tdata	Output	16	**Output Data:** **Packed IQ Format mode:** tdata[15:0] = I data[15:0] tdata[31:16] = Q data[15:0] **Unpacked IQ Format mode:** tdata[31:8] = I data[23:0], followed in time by Q data[23:0] (alternating). tdata[7:0] are zero
m_axis_pdti_tvalid		1	**Output Data Valid:** Asserted when data is valid on the **m_axis_pdti_tdata**. This signal will be asserted for each channel output sample in the multiplexed stream where the channel is enabled. In Unpacked IQ Format mode, each channel output takes two clock cycles for I and Q and therefore each enabled channel has and asserted tvalid for two consecutive clock cycles. Tready is not supported.
m_axis_pdti_tuser		128	**Output Side–Band data:** The Pentek PDTI definition defines the tuser signals in the following manner: tuser[63:0] = timestamp[63:0] tuser[64] = Gate tuser[72] = Sync tuser[80] = PPS tuser[91:88] = Samples/Cycle (is set to 1) tuser[92] = IQ of first sample in data 0 = I, 1 = Q Set to 0 for Packed IQ Format mode. Set to alternate 0 then 1 for I and Q data cycles for Unpacked IQ Format mode. tuser[94:93] = Data Format Set to 1 = 16–bit for Packed IQ Format mode. Set to 2 = 24–bit for Unpacked IQ Format mode. tuser[95] = Data Type (set to '1' = Complex) tuser[103:96] = channel[7:0] Marks the eight multiplexed output channels. Starts at the value specified by the generic parameter first_chan and increments up to first_chan+7 tuser[111:104] = saturation marker for each channel Each bit is asserted only during the corresponding channel data output. tuser[112] = Start of channel scan. Marks channel 0 in the eight channel scan. Note, channel 0 may not be enabled and therefore this mark will occur with tvalid not asserted. tuser[127:113] = Reserved

3.3 I/O Signals

The I/O port/signal descriptions of the top level module of the 8–Channel DDC core are provided in Table 3–4..

Table 3–4: I/O SIgnals

Port/Signal Name	Type	Direction	Description
Clock Signals			
axis_aclk	std_logic	Input	**AXI4-Stream clock for input and output AXI4-Stream data streams.**
s_axi_csr_aclk	std_logic	Input	**AXI-Lite clock for AXI-Lite bus. This is usually the PCIe clock at 250 MHz.**
Data Signals			
sat[7:0]	std_logic_ vector	Output	**Saturation signal:** Each channel, 0 through 7 has a corresponding saturation output bit. This marks saturation in the output gain stage.
Control Signals			
s_axi_csr_aresetn	std_logic	Input	**AXI-Lite Bus Reset:** Synchronous with s_axi_csr_aclk clock. Active low. This reset will reset all of the control registers. It will not clear coefficient set data or tuning frequencies.
axis_aresetn	std_logic	Input	**AXI-Lite Bus Reset:** Synchronous with axis_aclk clock. Active low. This reset resets the core logic, however it is not sufficient to commence operation. A sync signal must be received after the core is set up in order to begin proper filter operation.

This page is intentionally blank

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the **px_8ch_channelizer_ddc**. The memory map master table is provided in Table 4–1. Table 4–2 through Table 4–8 provide further details.

4.1 Memory Maps.

Table 4–1: Register Space Memory Map: Master Table

Register Name	Address (Base Address +)	Access	Description
Section 1 Registers (Table 4–2)	0x0000	R/W	Control/Status Registers for Section 1 of the Core.
Section 2 Registers (Table 4–3)	0x4000	R/W	Control/Status Registers for Section 2 of the Core.
Section 2 Tuning Frequency Tables (Table 4–4)	0x4800	R/W	Phase Offset Tables
Section 2 Phase Offset Tables (Table 4–5)	0x4C00	R/W	Phase Offset Tables
Section 3 Registers (Table 4–6)	0x8000	R/W	Control/Status Registers for Section 3 of the Core.
Section 3 Coefficient Memory (Table 4–7)	0xA000	R/W	Coefficient Memory for Section 3 Decimating FIR.
Section 4 Registers (Table 4–8)	0xC000	R/W	Control/Status Registers for Section 4 of the Core.

Table 4–2: Register Space Memory Map: Section 1 Control Registers

Register Name	Address (Base Address +)	Access	Description
Reset Register	0x00	R/W	Reset Register. Resets section 1 logic.
SYNC Enable Register	0x04	R/W	Sync Enable

4.1 Memory Maps (continued)

Table 4−3: Register Space Memory Map: Section 2 Control Registers

Register Name	Address (Base Address +)	Access	Description
Reset Register	0x00	R/W	Reset Register. Resets section 2 logic.
Control Register	0x04	R/W	Control Register for section 2

Table 4−4: Register Space Memory Map: Section 2 Tuning Frequency Table

Register Name	Address (Base Address +)	Access	Description
Tuning Frequency Set 0	0x000−0x01C	R/W	Set 0 of 8 32−bit Tuning Frequencies
Tuning Frequency Set 1	0x020−0x03C	R/W	Set 1 of 8 32−bit Tuning Frequencies
Tuning Frequency Set 2	0x040−0x05C	R/W	Set 2 of 8 32−bit Tuning Frequencies
Tuning Frequency Set 3	0x060−0x07C	R/W	Set 3 of 8 32−bit Tuning Frequencies

Table 4−5: Register Space Memory Map: Section 2 Phase Offset Table

Register Name	Address (Base Address +)	Access	Description
Phase Offset Set 0	0x000−0x01C	R/W	Set 0 of 8 32−bit Phase Offsets
Phase Offset Set 1	0x020−0x03C	R/W	Set 1 of 8 32−bit Phase Offsets
Phase Offset Set 2	0x040−0x05C	R/W	Set 2 of 8 32−bit Phase Offsets
Phase Offset Set 3	0x060−0x07C	R/W	Set 3 of 8 32−bit Phase Offsets

4.1 Memory Maps (continued).

Table 4–6: Register Space Memory Map: Section 3 Registers

Register Name	Address (Base Address +)	Access	Description
Control Registers			
Control Register	0x00	R/W	Control Register
Coefficient Load Register	0x04	R/W	Coefficient Load Register
Gain Register 1	0x08	R/W	Gain Control for CH0 and CH1
Gain Register 2	0x0C	R/W	Gain Control for CH2 and CH3
Gain Register 3	0x10	R/W	Gain Control for CH4 and CH5
Gain Register 4	0x14	R/W	Gain Control for CH6 and CH7
Status Registers			
Status Register	0x18	RO	Status Register
Interrupt Registers			
Interrupt Enable Register	0x1C	R/W	Interrupt Enable Register
Interrupt Status Register	0x20	RO	Interrupt Status Register
Interrupt Flag Register	0x24	R/CLR	Interrupt Flag Register

Table 4–7: Register Space Memory Map: Section 3 Coefficient Set Memory

Register Name	Address (Base Address +)	Access	Description
Coefficient Set	0x0000–0x3FFF	R/W	Coefficient Set. Load first half of symmetric filter. There should be (28xdecimation)/2 number of entries.

Table 4–8: Register Space Memory Map: Section 4 Control Registers

Register Name	Address (Base Address +)	Access	Description
Control Register	0x00	R/W	Control Register for section 4.

4.2 Section 1: Reset Register

This register is used to reset the logic in Section 1 of the core. This register is illustrated in Figure 4−1 and described in Table 4−9.

NOTE: The bits [8:0] are set based on the generic parameter initial_tap_delay defined by the user, which is used as the default value. The tap size may vary over temperature.

Figure 4−1: Reset Register 1

Table 4−9: Section 1: Reset Register (Base Address + 0x0000)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:1	Reserved	N/A	R/W	**Reserved**
0	Reset	0	R/W	Reset. Active high

4.3 Section 1: Sync Enable Register

This register is used to control the initialization of Section 1. This register is illustrated in Figure 4–2 and described in Table 4–10.

NOTE: The bits [31:16] and [6:2] are set based on the generic parameters defined by the user, which are used as default values.

Figure 4–2: Section 1: Sync Enable Register

Table 4–10: Section 1: Sync Enable Register (Base Address + 0x0004)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:1	Reserved	–	R/W	Reserved
0	Sync Enable	0	R/W	**SYNC Enable:** When asserted high, Section 1 of the core will be initialized for operation upon receipt of the SYNC input signal. When low, SYNC signals are ignored.

4.4 Section 2: Reset Register

This register is used to reset the logic in Section 2 of the core. This register is illustrated in Figure 4–3 and described in Table 4–11.

NOTE: The bits [31:0] are set based on the generic parameters defined by the user, which are used as default values.

Figure 4–3: Section 2: Reset Register

Table 4–11: Section 2: Reset Register (Base Address + 0x4000)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	–	R/W	Reserved
0	Reset	0	R/W	Reset. Active high

4.5 Section 2: Control Register

This register is used to control the initialization and frequency and phase control of Section 2. This register is illustrated in Figure 4–4 and the bits are described in Table 4–12.

Figure 4–4: Section 2: Control Register

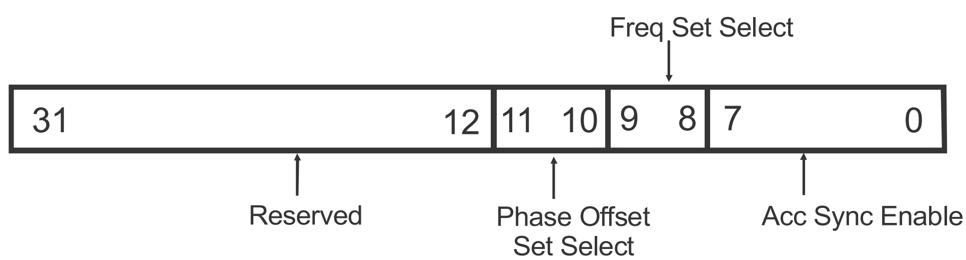


Table 4–12: Section 2: Control Register (Base Address + 0x4004)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:12	Reserved	–	R/W	**Reserved**
11:10	Phase Offset Set Select	“00”	R/W	Phase Offset Table Select. 00 = Table 0 selected 01 = Table 1 selected 10 = Table 2 Selected 11 = Table 3 selected
9:8	Freq Set Select	“00”	R/W	Frequency Table Select. 00 = Table 0 selected 01 = Table 1 selected 10 = Table 2 Selected 11 = Table 3 selected
7:0	Acc Sync Enable	0x00	R/W	**NCO Phase Accumulator SYNC Enable:** When a bit is asserted high, the corresponding channel's NCO Phase Accumulator is reset when a SYNC pulse is input to the core in the input AXI4–Stream data stream.

4.6 Section 2: Tuning Frequency Table Registers

These registers are used to control the tuning frequency of the eight channels. The set of eight frequencies selected is controlled by the Freq Set Select control bits in the Section 2 Control register. These registers are illustrated in Figure 4–5 and described in Table 4–13.The tuning frequency is calculated as (freq[31:0]/0x100000000) * axis_aclk rate.

Example:
axis_aclk = 200 MHz
freq[31:0] = 0x40000000
tuning frequency = 50 MHz

Figure 4–5: Tuning Frequency Table Registers

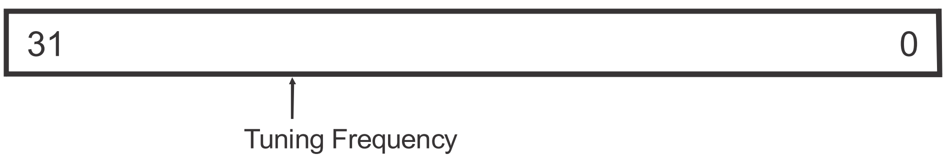


Table 4–13: Section 2: Tuning Frequency Table Registers				
(Base + 0x4800+ (Chan *4)+(Freq Set Select*32)) Where Chan = Channel number 0–7 Where Freq Set Select = Frequency Table Select 0–3				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:0	freq	0x0000	R/W	Tuning Frequency

4.7 Section 2: Phase Offset Tables

These registers are used to control the NCO Phase Offset of the eight channels. The set of eight phase offsets selected is controlled by the Phase Offset Set Select control bits in the Section 2 Control register. These registers are illustrated in Figure 4–6 and the bits are described in Table 4–14.

The phase offset is calculated as (phs_offset[31:0]/0x100000000) x 2π radians.

Figure 4–6: Phase Offset Table Registers

Table 4–14: Phase Offset Table Registers

(Base + 0x4C00+ (Chan *4)+ (Phase Offset Set Select*32))
Where Chan = Channel number 0–7
Where Phase Offset Set Select = Phase Offset Set Select 0–3

Bits	Field Name	Default Value	Access Type	Description
31:0	phs_offset	0x000 0	R/W	**Phase Offset**

4.8 Section 3: Control Register

This register is used to control operation of the Section 3 decimating filter. This register is illustrated in Figure 4−7 and described in Table 4−15.

Figure 4−7: Section 3: Control Register

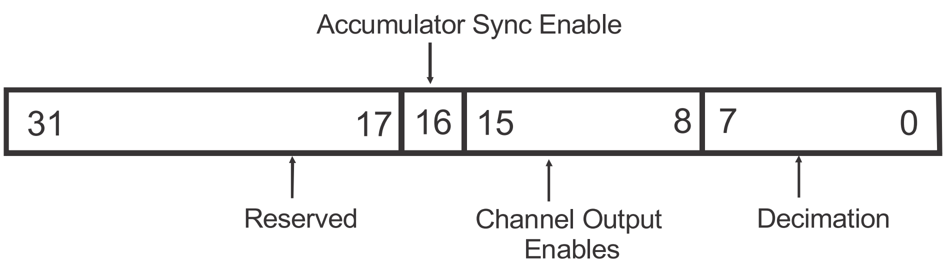


Table 4−15: Section 3: Control Register (Base + 0x8000)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:17	Reserved	–	R/W	**Reserved**
16	Acc Sync Enable	0	R/W	**FIR Accumulator Sync Reset Enable:** When asserted, the filter accumulator and output decimator are aligned and synchronized to the data stream when a SYNC pulse is received in the input data stream. This must be done in conjunction with the Sync Enables in Section 1 and 2 after initial setup and any time the decimation setting is changed. It affects all eight channels. When de−asserted, SYNC pulses are ignored.
15:8	Chan En	0xFF	R/W	**Channel Output Enables:** To enable a channel onto the core output stream, set its corresponding channel enable bit to a '1'. In the output AXI4−Stream data stream, tvalid is asserted only for enabled channels in the multiplexed stream.
7:0	Dec	0x01 (Decimate by 2)	R/W	**Decimation:** This controls the decimation value of the FIR. It can be set to any value from 0 to 127 corresponding to decimations of 2 to 127. **Note:** The overall decimation is equal to 8*(Dec+1), or 8 times the decimation of the Section 3 FIR. Changing decimation requires an enabled SYNC pulse to align the FIR accumulator and decimator.

4.9 Section 3: Coefficient Load Register

This register is used to control loading of the coefficients into the filter. This register is illustrated in Figure 4–8 and described in Table 4–16.

Figure 4–8: Section 3: Coefficient Load Register

Table 4–16: Section 3: Coefficient Load Register (Base + 0x8004)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:1	Reserved	–	R/W	**Reserved**
0	Coef_Load	0	R/W	To load the coefficients from the coefficient set memory to the FIR filter, this bit must be toggled from a '0' to a '1' and back to '0'. Coefficients must first be written into the Coefficient Set Memory. Also, the desired decimation value must be set before toggling this bit. Once the load process begins, completion of the load process can be monitored in the Section Status Register.

4.10 Section 3: Gain Control Register 1

This register is used to control output gain after the filter for channels 0 and 1. This register is illustrated in Figure 4–9 and described in Table 4–17.

Figure 4–9: Section 3: Gain Control Register 1

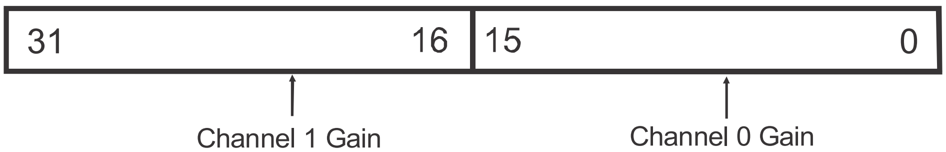


Table 4–17: Section 3: Gain Control Register 1 (Base + 0x8008)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:16	CH1 Gain[15:0]	0x0800	R/W	Channel 1 Gain
15:0	CH0 Gain[15:0]	0x0800	R/W	Channel 0 Gain

NOTE: Assuming use of the coefficient sets provided by Pentek, a roughly unity gain can be achieved by setting GAIN[15:0] to 5461/(DEC[7:0]+1).
GAIN[15:0] is an unsigned integer.

4.11 Section 3: Gain Control Register 2

This register is used to control output gain after the filter for channels 2 and 3. This register is illustrated in Figure 4–10 and described in Table 4–18.

Figure 4–10: Section 3: Gain Control Register 2

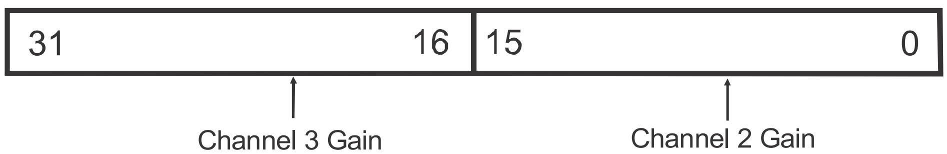


Table 4–18: Section 3: Gain Control Register 2 (Base + 0x800C)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:16	CH3 Gain[15:0]	0x080 0	R/W	Channel 3 Gain
15:0	CH2 Gain[15:0]	0x080 0	R/W	Channel 2 Gain

NOTE: Assuming use of the coefficient sets provided by Pentek, a roughly unity gain can be achieved by setting GAIN[15:0] to 5461/(DEC[7:0]+1).
GAIN[15:0] is an unsigned integer.

4.12 Section 3: Gain Control Register 3

This register is used to control output gain after the filter for channels 4 and 5. This register is illustrated in Figure 4–11 and described in Table 4–19.

Figure 4–11: Section 3: Gain Control Register 3

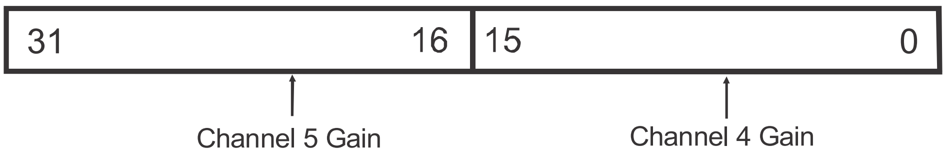


Table 4–19: Section 3: Gain Control Register 3 (Base + 0x8010)

Bits	Field Name	Default Value	Access Type	Description
31:16	CH5 Gain[15:0]	0x0800	R/W	Channel 5 Gain
15:0	CH4 Gain[15:0]	0x0800	R/W	Channel 4 Gain

NOTE: Assuming use of the coefficient sets provided by Pentek, a roughly unity gain can be achieved by setting GAIN[15:0] to 5461/(DEC[7:0]+1). GAIN[15:0] is an unsigned integer.

4.13 Section 3: Gain Control Register 4

This register is used to control output gain after the filter for channels 6 and 7. This register is illustrated in Figure 4–12 and described in Table 4–20.

Figure 4–12: Section 3: Gain Control Register 4

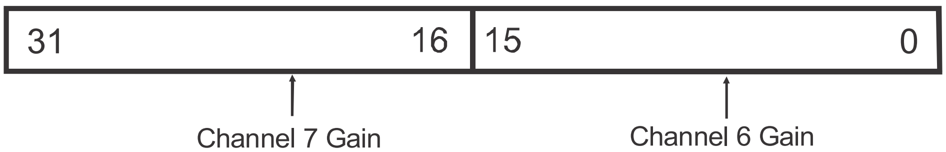


Table 4–20: Section 3: Gain Control Register 4 (Base + 0x8014)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:16	CH7 Gain[15:0]	0x0800	R/W	Channel 7 Gain
15:0	CH6 Gain[15:0]	0x0800	R/W	Channel 6 Gain

NOTE: Assuming use of the coefficient sets provided by Pentek, a roughly unity gain can be achieved by setting GAIN[15:0] to 5461/(DEC[7:0]+1).
GAIN[15:0] is an unsigned integer.

4.14 Section 3: Status Register

This register is used to read back status of the coefficient load execution. This register is illustrated in Figure 4–13 and described in Table 4–21.

Figure 4–13: Section 3: Status Register

Table 4–21: Section 3: Status Register (Base + 0x8018)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:2	Reserved	–	RO	Reserved
1	Load Done	–	RO	**Load Busy:** When asserted high, the coefficient load is still in process.
0	Load Busy	–	RO	**Load Done:** When asserted high, the coefficient load is complete.

4.15 Section 3: Interrupt Enable Register

This register is used to enable specific interrupt sources. This register is illustrated in Figure 4–14 and described in Table 4–22.

Figure 4–14: Section 3: Interrupt Enable Register

Table 4–22: Section 3: Interrupt Enable Register (Base + 0x801C)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:8	Reserved	–	R/W	Reserved
7:0	Sat En[7:0]	–	R/W	**Saturation Interrupt Enable[7:0]:** When set to '1', a saturation occurrence in the output gain stage of the corresponding channel will create an interrupt.

4.16 Section 3: Interrupt Status Register

This register shows the current value of saturation on each channel. Since the value is transitory, this is of little use. Instead, use the Interrupt Flag register to read if saturation has occurred. This register is illustrated in Figure 4−15 and described in Table 4−23.

Figure 4−15: Section 3: Interrupt Status Register

Table 4−23: Section 3: Interrupt Status Register (Base + 0x8020)

Bits	Field Name	Default Value	Access Type	Description
31:8	Reserved	–	RO	Reserved
7:0	Sat[7:0]	–	RO	**Saturation Interrupt[7:0]:** When set to '1', a saturation occurrence in the output gain stage of the corresponding channel is happening.

4.17 Section 3: Interrupt Flag Register

This register shows the latched value of saturation on each channel. This register is illustrated in Figure 4–16 and described in Table 4–24.

Figure 4–16: Section 3: Interrupt Flag Register

Table 4–24: Section 3: Interrupt Flag Register (Base + 0x8024)				
Bits	**Field Name**	**Default Value**	**Access Type**	**Description**
31:8	Reserved	–	RO	Reserved
7:0	Sat Flag[7:0]	–	R/CLR	**Saturation Interrupt Flag[7:0]:** When set to '1', a saturation occurrence in the output gain stage of the corresponding channel has been latched. Clear by writing a '1' to the bits you desire to clear.

4.18 Section 3: Coefficient Set

This memory space is used to store the coefficient set for the decimating FIR filter. This is described in Table 4–25.

NOTE: For any given decimation setting, a filter coefficient set that is even symmetrical and of a length of 28*(DEC[7:0]+1) or 28 times the decimation of the FIR must be used. Write only the first half of the symmetrical filter coefficient set into this memory space. The coefficients will not be loaded into the filter until the Coefficient Load Control bit has been toggled. Make sure the correct decimation setting has been written to the Section 3 Control Register before loading the coefficients to the filter.

The values are signed 18–bit integers, but the memory is 32–bit words. Sign extend the 18–bit value to a 32–bit value.

Table 4–25: Section 3: Coefficient Set

(Base + 0xA000+ (Coef Num*4))
Where Coef Num is the coefficient number from 0 to ((28*(DEC[7:0]+1)/2)–1)

Bits	Field Name	Default Value	Access Type	Description
31:18	Not used	–	R/W	Not used.
17:0	Coef	–	R/W	**Coefficient:** The coefficients are 18–bit signed values. Only the lower 18 bits of the 32–bit value are used by the filter, but good practice would be to sign extend the 18–bit value to a full 32–bit value.

4.19 Section 4: Control Register

This register controls the output data format. This register is illustrated in Figure 4–17 and described in Table 4–26.

Figure 4–17: Section 4: Control Register

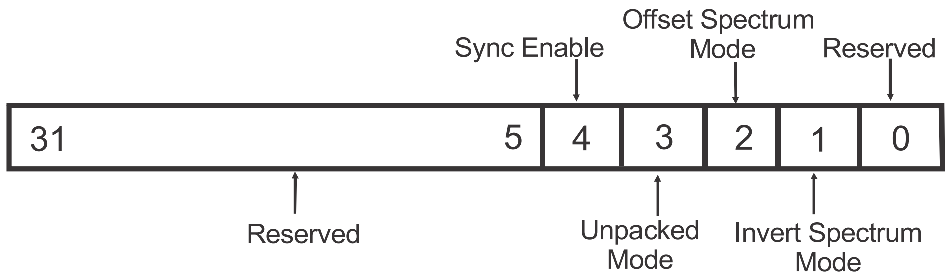


Table 4–26: Section 4: Control Register (Base + 0xC000)

Bits	Field Name	Default Value	Access Type	Description
31:8	Reserved	–	R/W	Reserved
4	Sync Enable	‘0’	R/W	**Sync Enable:** This bit when asserted allows the output data formatter to be synchronized by a received SYNC signal. This only has an effect in Offset Spectrum Mode.
3	Unpacked Mode	‘0’	R/W	**Unpacked 24–bit Output Mode:** 0 = Packed IQ 16–bit mode Q[15:0], I[15:0] 1 = Unpacked IQ 24–bit mode 24 bits are left justified in the 32–bit output. I and Q are consecutive in the output stream. I[23:0],0x00 Q[23:0],0x00
2	Offset Spect	‘0’	R/W	**Offset Spectrum Mode:** The output data spectrum is offset around fo/4.
1	Invert Spect	‘0’	R/W	**Invert Spectrum Mode:** The output data spectrum is inverted.
0	Reserved	–	R/W	Reserved

This page is intentionally blank

Chapter 5: Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the px_8ch_channelizer_ddc.

5.1 General Design Guidelines

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

5.2 Clocking

AXI4–Lite Clock: `s_axi_csr_aclk`

The `s_axi_csr_aclk` is used to clock the AXI4–Lite Control/Status Register (`s_axi_csr`) interface of the core.

AXI4–Stream Interface Clock: `axis_aclk`

5.3 Resets

Main AXI4–Lite reset: `s_axi_csr_aresetn`

This is an active low synchronous reset associated with the s_axi_csr_aclk. When asserted, all control registers are reset to their default state.

Main AXI4–Stream reset: axis_aresetn

This is an active low synchronous reset associated with the axis_aclk. When asserted, all logic and state machines except the control registers are reset. This reset is not sufficient to start operation of the core. The core must first be set up with the desired modes and coefficients loaded. Then, with the sync enable control bits in each section asserted, a sync signal must be asserted on the input AXI4–Stream data stream. The sync will start the core functioning and can be used to synchronize multiple cores with each other. The eight NCOs can be independently masked to reset when sync is received and can be synchronized independently any time after main core operation commences is the sync enable bits for all other sections of the core are de–asserted.

Important: Proper operation of the core is not possible without a sync pulse on the input stream to initialize it. The sync pulse can be one clock cycle wide or wider. If it is wider, the initialization will take place on the rising edge of the sync signal.

5.4 Interrupts

This core has an edge type (rising edge–triggered) interrupt output. It is synchronous with the **s_axi_csr_aclk**. On the rising edge of any interrupt signal, a one clock cycle wide pulse is output from the core on it's **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 to create an interrupt.

NOTE: Because this core uses edge–triggered interrupts, the fact that an interrupt condition may remain 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

Control/Status Register Interface: This is the control/status register Interface. It is associated with the **s_axi_csr_aclk**. It is a standard AXI4–Lite type interface. See Chapter 4 for the control register memory map, for more details on the registers that can be accessed through this interface.

Input Stream Data (PDTI) Interface: This interface is used to transfer data into the core. It is a standard AXI4–Stream Slave Interface configured to be compatible with Pentek's PDTI type AXI4–Stream single sample per cycle data streams. This interface has no tready and does not support flow control. Further, it assumes that it will receive valid data on every clock cycle. Tvalid must be held constantly asserted. For more details about this interface refer to Table 3–2.

Output Stream Data (PDTI) Interface: This interface is used to transfer data out of the core. It is a standard AXI4–Stream Master Interface configured to be compatible with Pentek's PDTI type AXI4–Stream. The output is always complex data and 32–bits wide. In packed IQ mode, the output is 16–bits I and 16–bit Q packed into a 32–bit tdata word. In unpacked 24–bit mode, the data is consecutive I and Q data with 24–bits each left justified in the 32–bit tdata word. It is a channel multiplexed stream with up to eight output channels. Disabled channels in the stream have tvalid deasserted. Channel indication is embedded in the AXI4–Stream tuser signal as defined in the Pentek PDTI type specification. This interface has no tready and does not support flow control. For more details about this interface refer to Table 3–3.

5.6 Programming Sequence

The following steps must be followed in order to achieve proper core output:

1) Set up the desired decimation, tuning frequencies, phase offsets, channel enables, gains, and output data format controls.

2) Write the first half of the even symmetrical (28xdecimation) length filter coefficient set into the coefficient set memory.

3) Toggle the coefficient load control bit to initiate coefficient loading.

4) Wait until coefficient loading completes.

5) Assert the sync enable control bits in all core sections that have one.

6) Assert the SYNC bit in the input PDTI AXI4–Stream for at least one clock cycle.

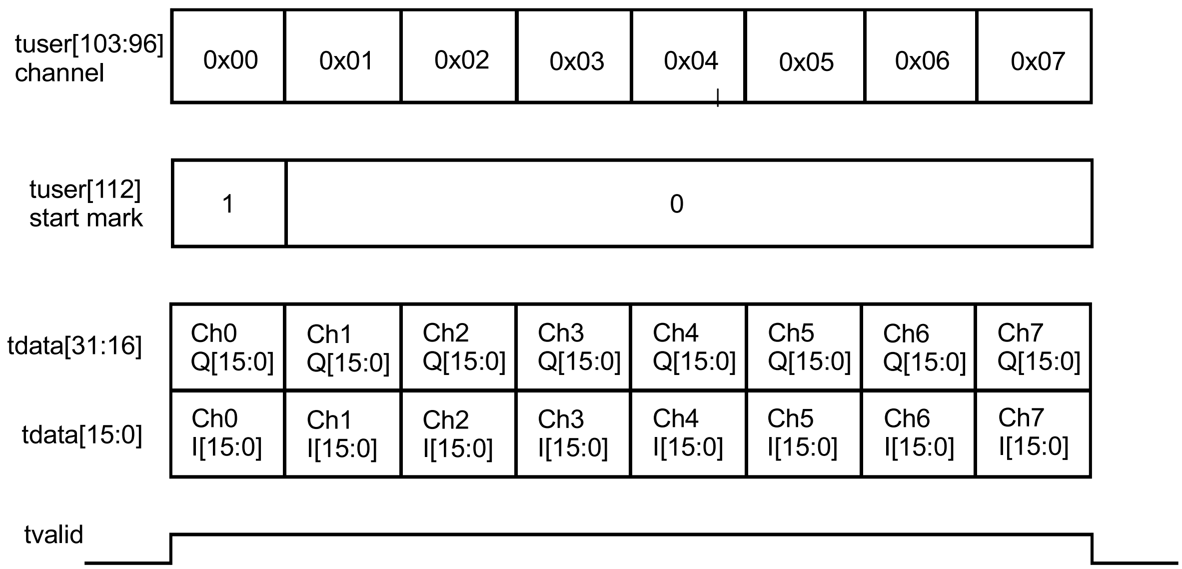
7) De–assert all sync enable control bits.

Any time that decimation is changed, the sync enable control bits must be asserted and a SYNC signal pulse must be sent to the core to realign internal logic.

Frequencies, phases, gains and channel enables can be changed on the fly. There is room to enter up to four frequency tables.

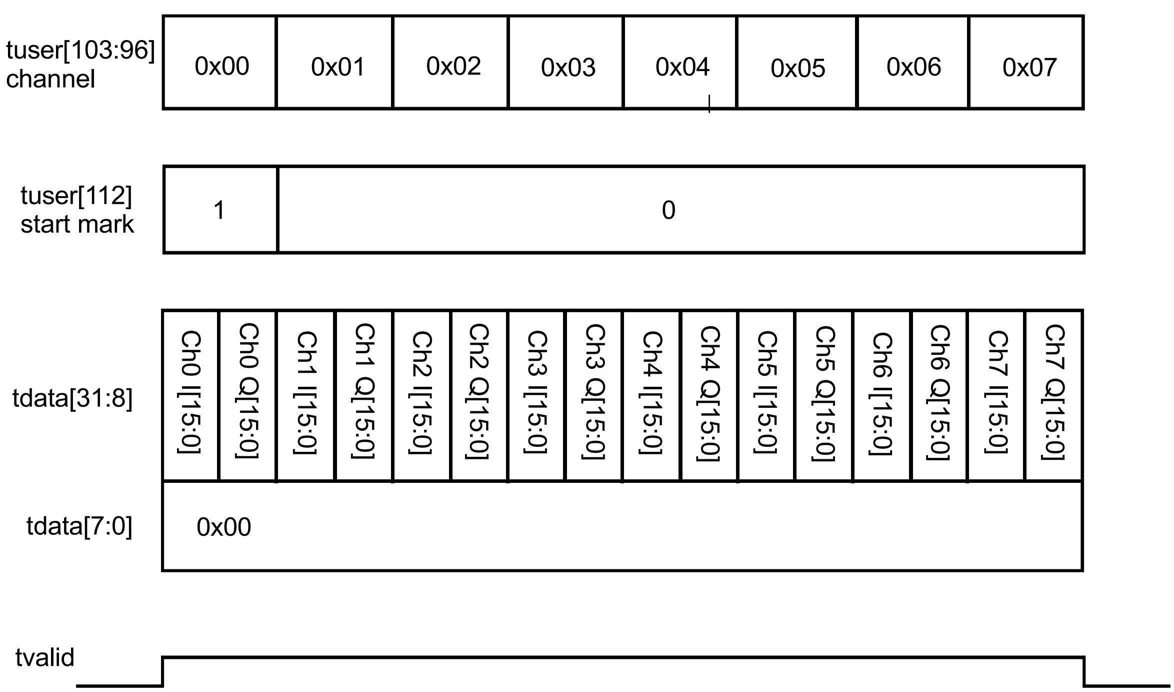
5.7 Timing Diagrams

Figure 5−1: Packed 16−bit IQ Output Mode Output AXI4−Stream



tvalid is asserted only during enabled channels. This diagram shows 8 channels enabled.

Figure 5−2: Figure 5−2: Un−Packed 24−bit IQ Output Mode Output AXI4−Stream



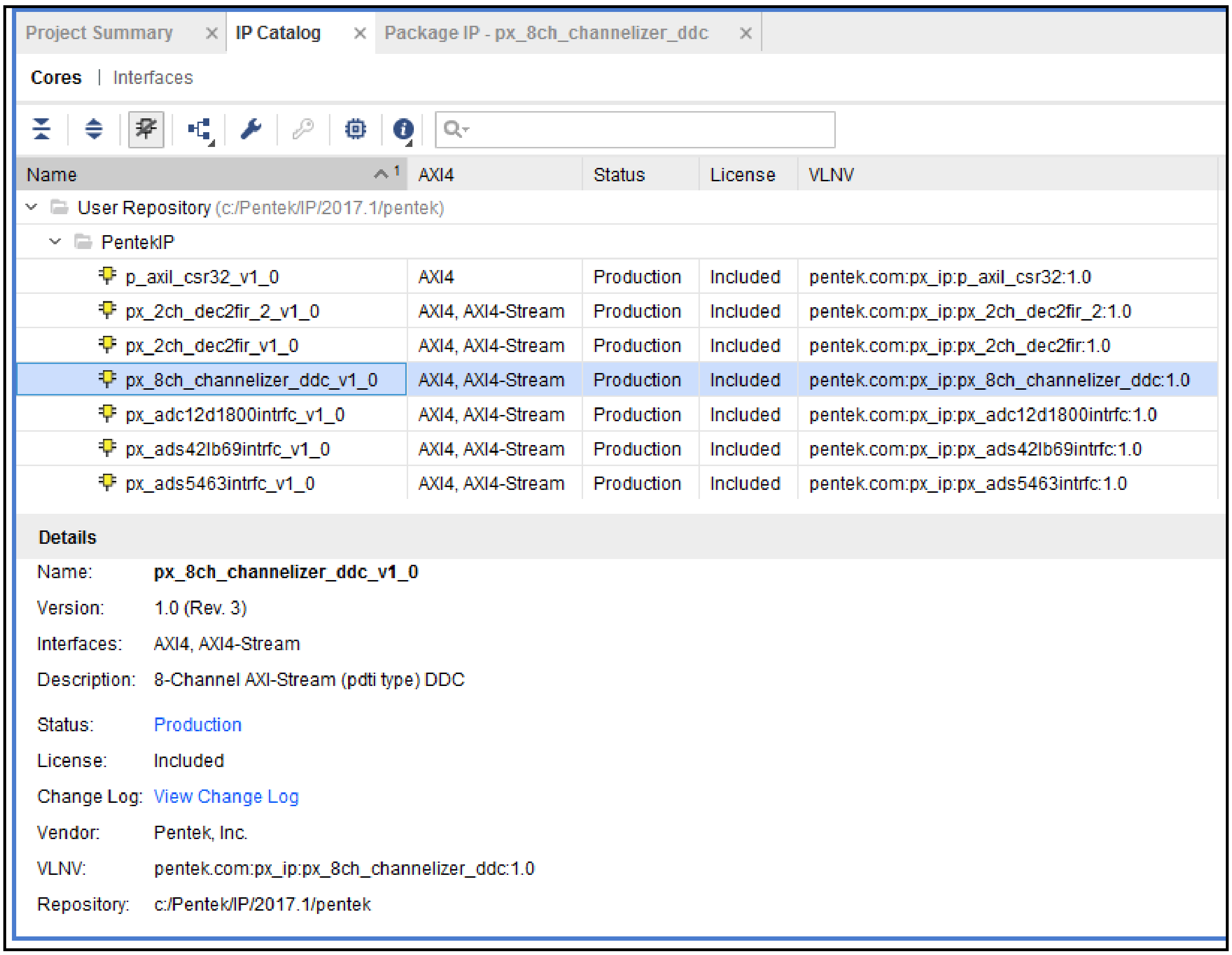
tvalid is asserted only during enabled channels. This diagram shows 8 channels enabled.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek 8–Channel DDC Core. It also includes simulation, synthesis, and implementation steps that are specific to this core. This IP 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_8ch_channelizer_ddc_v1_0** as shown in Figure 6–1.

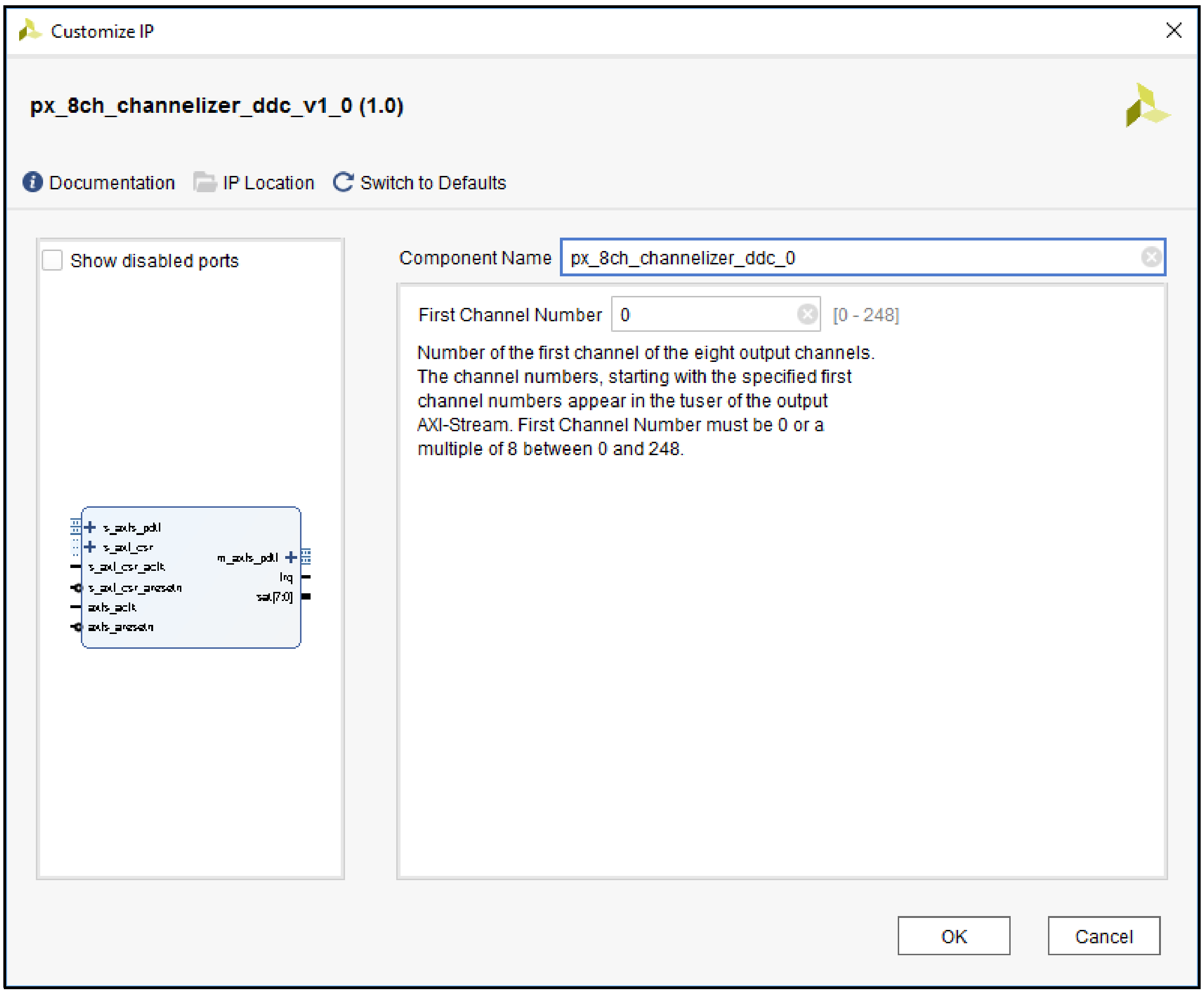
Figure 6–1: Pentek 8–Channel DDC Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_8ch_channelizer_ddc_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: Pentek 8–Channel DDC Core IP Symbol



6.2 User Parameters

6.2.1 First Channel Number

The output AXI4–Stream data stream contains channels numbers in tuser to identify the channel number. The eight output channels will be marked with consecutive channel numbers starting from the value set by the **FIRST CHANNEL NUMBER** parameter. The value of this parameter must be a value of 0 or a multiple of 8 up to a maximum of 248 for the core to operate properly. If you have multiple cores in a design, setting them to different channel number ranges will help identify the data source.

Example: First Channel Number is set to 248.

Output Channel 0 will have a channel number of 248

Output Channel 1 will have a channel number of 249

Output Channel 2 will have a channel number of 250

Output Channel 3 will have a channel number of 251

Output Channel 4 will have a channel number of 252

Output Channel 5 will have a channel number of 253

Output Channel 6 will have a channel number of 254

Output Channel 7 will have a channel number of 255

6.3 Generating Output

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 core in Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with this 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 Kintex Ultrascale FPGAs.

6.4 Constraining the Core (continued)

Clock Frequencies

The maximum **axis_aclk** frequency for this IP core is 350 MHz while the AXI4−Lite interface clock (**s_axi_csr_aclk**) frequency is 250 MHz.

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.

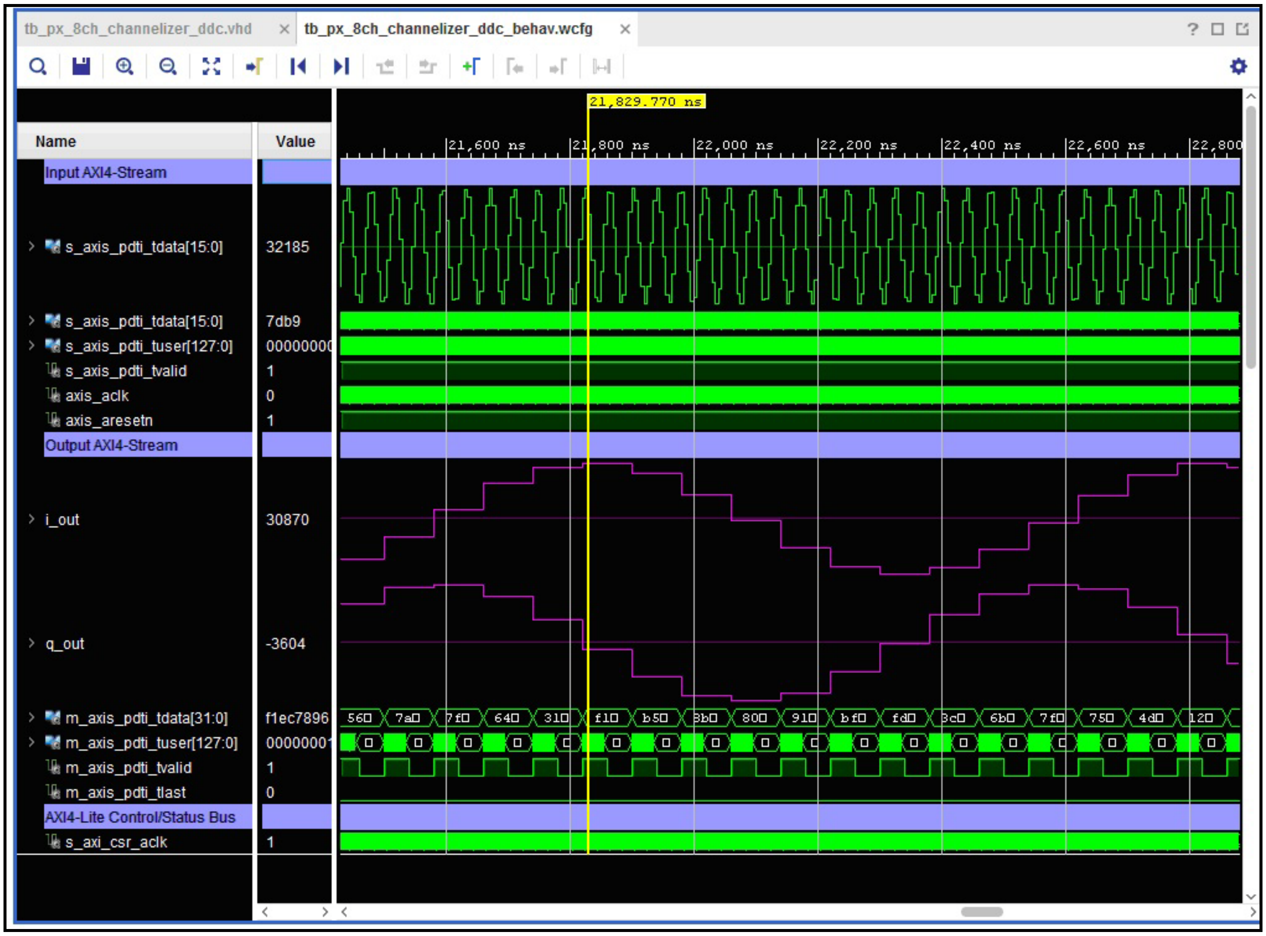
6.5 Simulation

The test bench provided allows you to demonstrate basic operation of the core. Several generic parameters are provided that will allow you to modify the behavior of the simulation. Not all possible features are exercised.

Table 6–1: Generic Parameters

Generic	Type	Description	Notes
clk_freq	real	AXI4–Stream sample clock frequency	In MHz. Must have decimal point. Example 200.0 for 200 MHz.
data_freq	real	Test Stimulus Sinewave input frequency	In MHz. Must have decimal point. Example 20.0 for 20 MHz.
tuning_freq_0	real	Channel 0 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_1	real	Channel 1 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_2	real	Channel 2 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_3	real	Channel 3 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_4	real	Channel 4 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_5	real	Channel 5 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_6	real	Channel 6 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
tuning_freq_7	real	Channel 7 Tuning Frequency	In MHz. Must have decimal point. Example 40.1 for 40.1 MHz.
enable_mask[7:0]	std_logic_vector	Channel Enables	Channel is enabled when its corresponding bit is set to '1'.
Decimation	integer	Decimation of Last Stage Decimating FIR.	Total decimation will be 8 times this setting. This test bench can only test values of 2,3,4,7,8, and 11.

Figure 6–3: Pentek 8–Channel DDC Core Behavior



6.6 Synthesis and Implementation

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