IP CORE MANUAL

AXI4–Stream VITA 49 Unpacketizer IP

px_vita49_unpkt

PENTEK

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

Manual Part Number: 807.48406 Rev: 1.1 – October 3, 2017

Manual Revision History

Date	Version	Comments
9/20/17	1.0	Initial Release
10/3/17	1.1	Revised Sect 1.1, Table 2–1, and Figure 6–6. Added new Figure 6–7.

Legal Notices

Copyright

Trademarks

Printed in the United States of America.

Table of Contents

Page

Table of Contents

Page

Chapter 4: Register Space

Chapter 5: Designing with the Core

Table of Contents

Page

Chapter 6: Design Flow Steps

Table of Contents

Page

This page is intentionally blank

IP Facts

Description

Pentek's Navigator™ AXI4–Stream VITA 49 Unpacketizer Core takes a packetized VITA 49 data stream and strips off the non–data portions of the packet leaving only the data in the outgoing stream.

This core complies with the ARM® AMBA® AXI4 Specification and also provides a control/status register interface. This user manual defines the hardware interface, software interface, and parameterization options for the AXI4–Stream VITA 49 Unpacketizer Core.

Features

- Core supports 256–bit wide VITA 49 input streams.
- Supports many types of VITA 49 packets and optional fields.
- Core automatically identifies required and optional non–data fields and removes them from the stream leaving only data as the output.
- Minimal setup required, just reset, enable and go.
- Option to ignore **`tlast`** as an input.

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 2017.2 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

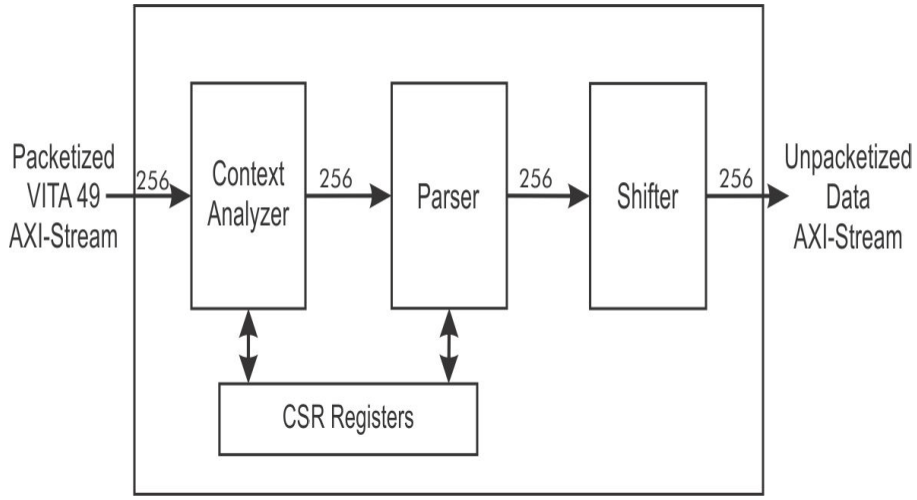
Chapter 1: Overview

1.1 Functional Description

The VITA 49 Unpacketizer Core (AXI4–Stream VITA 49 Unpacketizer Core) takes a packetized VITA 49 data stream and strips off the non–data portions of the packet leaving only the data in the outgoing stream.

Figure 1–1 is a top–level block diagram of this core. The modules within the block dia–gram are explained in the later sections of this user manual.

Figure 1–1: AXI4–Stream VITA 49 Unpacketizer Core Block Diagram



- **Context Analyzer:** This module analyzes the packet header and identifies the length of the packet and which optional fields are used in each packet.
- **Parser:** This module takes the length of the packet along with the fields identified by the context analyzer and labels all non–data fields. Then this core outputs the remaining data stream.
- **CSR Registers:** This module contains the control and status registers including Interrupt Enable, Interrupt Flag, and Interrupt Status registers. Registers are accessed through the AXI4– Lite interface.
- **Shifter:** This module shifts all the fields identified by the parser out of the data stream. It leaves the remaining data intact.

1.2 Applications

This core can be used with the Pentek VITA 49 Packetizer Core (**px_vita49_pkt**) to remove all non–data fields from a VITA 49 packet. This core could also be used with any valid VITA 49 stream coming from a VITA 49 compliant device which uses an AXI–Stream bus as an output.

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

4) VITA Radio Transport (VRT) Draft Standard
http://shop.vita.com/ANSI–VITA–490–2015–VITA–Radio–Transport–VRT–Standard–AV490.htm

Chapter 2: General Product Specifications

2.1 Standards

The AXI4–Stream VITA 49 Unpacketizer Core has bus interfaces that comply with the *ARM AMBA AXI4–Lite Protocol Specification* and the *AMBA AXI4–Stream Protocol Specification*. This core also complies with VITA–49.0 – 2015.

2.2 Performance

The emergence of high–speed interconnects enables the transmission of signals, such as digitized IF, over packet networks. However, packetization comes at the expense of some overhead. The VRT protocol can minimize this overhead by allowing the volume of data, the format of data, and the type of data to be configured for optimal link utilization. Thus the overhead of the protocol can typically be configured to be a small fraction of the overall signal data bandwidth, providing the same efficiency as proprietary implementations. This core is able to support any valid packet length and optional VITA 49 fields as long as there is not more than one header every 256 bits.

2.3 Resource Utilization

The resource utilization of the VITA 49 Unpacketizer Core 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
LUTs	1616
Flip–Flops	1233

NOTE: Actual utilization may vary based on the user design in which the VITA 49 Unpacketizer Core is incorporated.

2.4 Limitations and Unsupported Features

- This core does not support more than one header every 256 bits.
- This core only supports 256–bit input and output streams.

2.5 Generic Parameters

This section is not applicable to this IP core.

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

3.1 AXI4–Lite Core Interfaces

The AXI4–Stream VITA 49 Unpacketizer Core uses the Control/Status Register (CSR) interface to control, and receive status 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 VITA 49 Unpacketizer 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 signal 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 VITA 49 Unpacketizer Core. Note that the Register Space registers occupy an address range of [Base Address + (0x00 to 0x1C)].
s_axi_csr_awprot	Input	3	**Protection:** The VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer Core when it is ready to accept the read address. The address is latched when **s_axi_csr_arvalid** and **s_axi_csr_ aready** 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 VITA 49 Unpacketizer 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 VITA 49 Unpacketizer 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.

3.2 AXI4–Stream Core Interface

The Unpacketizer Core has the following AXI4–Stream Interface, used to receive and transfer data streams.

3.2.1 Stream Data Interface

This interface is used to transfer AXI4–Stream through the input and output ports of the Unpacketizer Core. Table 3–2 defines the ports in the Stream Data Interface.

Table 3-2: Stream Data Interface Interface Port Descriptions

Port	Direction	Width	Description
axis_aclk	Input	1	**Clock for the core**
axis_aresetn	Input	1	**Reset for the core:** Active low
s_axis_tvalid	Input	1	**Input Data Valid**
s_axis_tready	Output	1	**'1' when core is ready to accept data**
s_axis_tdata	Input	256	**VITA 49 packed stream**
s_axis_tkeep	Input	31	**tkeep for tdata:** Each bit corresponds to 8 bits of **`tdata`**.
s_axis_tlast	Input	1	**tlast for tdata**
m_axis_tvalid	Output	1	**tvalid for output data stream**
m_axis_tready	Input	1	**tready for master bus**
m_axis_tdata	Output	256	**VITA 49 data only**
m_axis_tkeep	Output	32	**tkeep for tdata:** Each bit corresponds to 8 bits of **`tdata`.**
m_axis_tlast	Output	1	**tlast for the master bus**

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the AXI4–Stream VITA 49 Unpacketizer Core. The memory map is shown in Table 4–1.

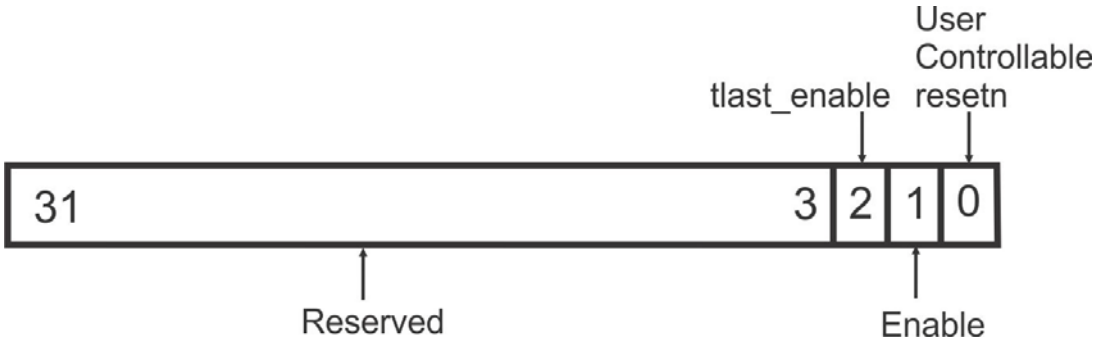
Table 4–1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Control Registers			
User Control	0x00	R/W	User Controls for: Bit 0: User Reset Bit 1: User Enable Bit 2: `Tlast` Enable
Interrupt Enable/Status/Flag Registers			
Interrupt Enable Register	0x04	R/W	Interrupt Enable Bits: Bit 0: Packet count error
Interrupt Status Register	0x08	RO	Interrupt Status Bits: Bit 0: Packet count error
Interrupt Flag Register	0x0C	R/Clr	Interrupt flag bits: Bit 0: Packet count error

4.1 User Control Register

This register is used to control the user control register. This core can toggle the user controllable reset and enable registers. The user reset register provides a easy user programmable reset for debug. The enable enables the core and halts the core if set to '0'. The User Control Register is illustrated in Figure 4–1 and described in Table 4–2.

Figure 4–1: User Control Register



.

Table 4–2: User Control Register (Base Address + 0x00)

Bits	Field Name	Default Value	Access Type	Description
0	user_resetn	0	R/W	**User controllable `resetn`:** Active low. Resets entire core. Clears all internal registers and packet information.
1	enable	0	R/W	**Enable:** The core will stop all processes when this bit is set to '0.' The core will function normally when this bit is set to '1.' **0 = not enabled** **1 = enabled**
2	tlast_enable	0	R/W	**`Tlast` Enable:** The core will ignore any **`tlast`** inputs unless this enable is turned on. **0 = not enabled** **1 = enabled**
3:31	Reserved	–	–	**Reserved**

4.2 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 4.5). This register is illustrated in Figure 4–2 and described in Table 4–3.

Figure 4–2: Interrupt Enable Register

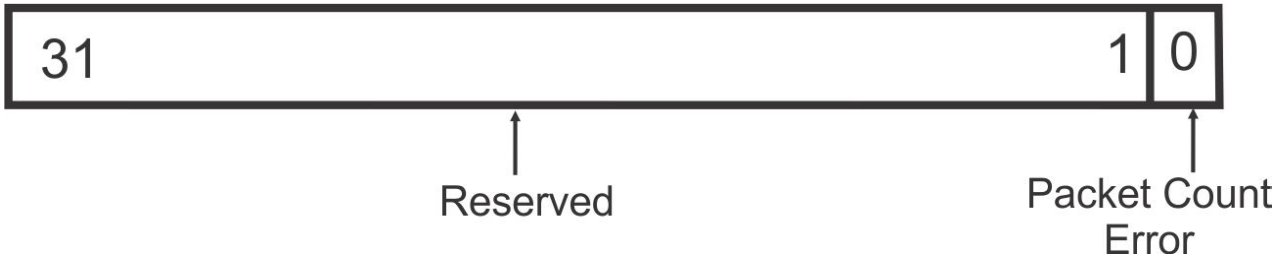


Table 4–3: Interrupt Enable Register (Base Address + 0x04)

Bits	Field Name	Default Value	Access Type	Description
0	packet_count_error	0	R/W	**Interrupt Enable for Packet Count Error:** This will go off when the packet count between packets does not increase by one. This means that a packet was lost.
31:1	Reserved	–	–	**Reserved**

4.3 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. This register is illustrated in Figure 4–3 and described in Table 4–4.

Figure 4–3: Interrupt Status Register

Table 4–4: Interrupt Status Register (Base Address + 0x08

Bits	Field Name	Default Value	Access Type	Description
0	packet_count_error	0	RO	**Interrupt Status for Packet Count Error:** This will go off when the packet count between packets does not increase by one. This means that a packet was lost.
31:1	Reserved	–	–	**Reserved**

4.4 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.This register is illustrated in Figure 4–4 and described in Table 4–5.

Figure 4–4: Interrupt Flag Register

Table 4–5: Interrupt Flag Register (Base Address + 0x0C)

Bits	Field Name	Default Value	Access Type	Description
0	packet_count_error	0	R/CLR	**Flag Status for Packet Count Error:** This will go off when the packet count between packets does not increase by one. This means that a packet was lost.
31:1	Reserved	–	–	**Reserved**

This page is intentionally blank

Chapter 5: Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the AXI4−Stream VITA 49 Unpacketizer Core.

5.1 General Design Guidelines

The AXI4−Stream VITA 49 Unpacketizer Core takes a VITA 49 packet, analyzes the header contents for optional fields, and strips them from the stream leaving only the data field as the output. The limitation of this core is that there cannot be more than one header per each 256−bit word.

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**

This clock is used to clock the AXI4−Stream inputs and outputs of the core.

5.3 Resets

Main reset: **axis_aresetn**

This is an active low synchronous reset associated with the **axis_aclk.** When asserted, all state machines in the core are reset, all FIFOs are flushed and all the control registers are cleared back to their initial default states.

CSR Reset: **s_axi_csr_resetn**

This is an active low sync reset associated with **s_axi_csr_aclk**. When asserted all con−trol registers, initial values, and interrupts are reset.

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 **irq** output. Each interrupt event is stored in two registers, accessible on the **s_axi_csr** bus.

5.4 Interrupts (continued)

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

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

Stream Data AXI Interface: This interface is used to transfer the input VITA 49 data stream and the data–only output stream. It is a standard AXI4–Stream Slave and Master Interface. For more details about this interface refer to Section 3.2.

5.6 Programming Sequence

This section briefly describes the programming sequence of registers in the VITA 49 Unpacketizer Core.

1) Use hardwired **aresetn**. Set to '1.'

2) Incoming VITA 49 data stream should not yet be active.

3) Set user reset and enable registers. The Control Register Address is 0. Set it to 0x00000003. This sets the user reset (bit 0) and the user enable (bit 1) to '1.' The core is now ready to use.

4) If desired enable **tlast_enable** to examine **tlast**. Otherwise incomming **tlast** will be ignored.

5) Send VITA 49 packets to core and the data–only portion of the packets will be output.

5.7 Timing Diagrams

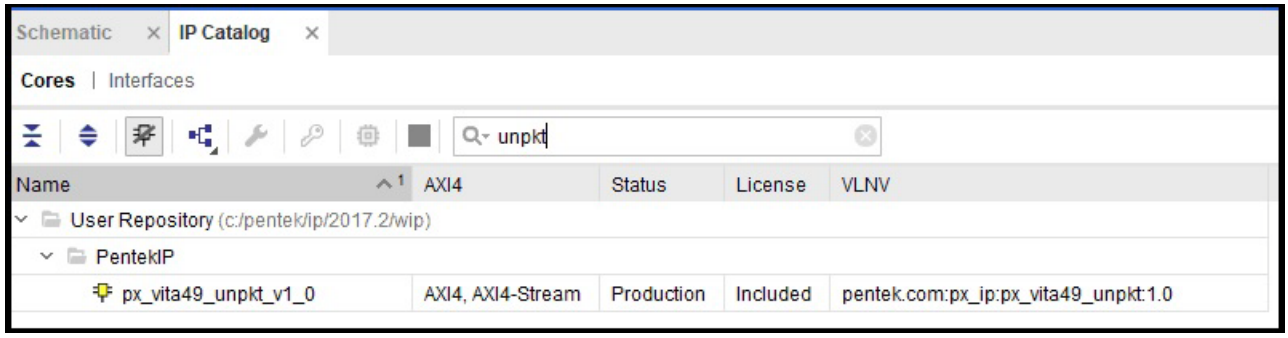
For more details about the test bench, refer to Section 6.5.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek AXI4–Stream VITA 49 Unpacketizer 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_vita49_unpkt_1_v1_0** as shown in Figure 6–1.

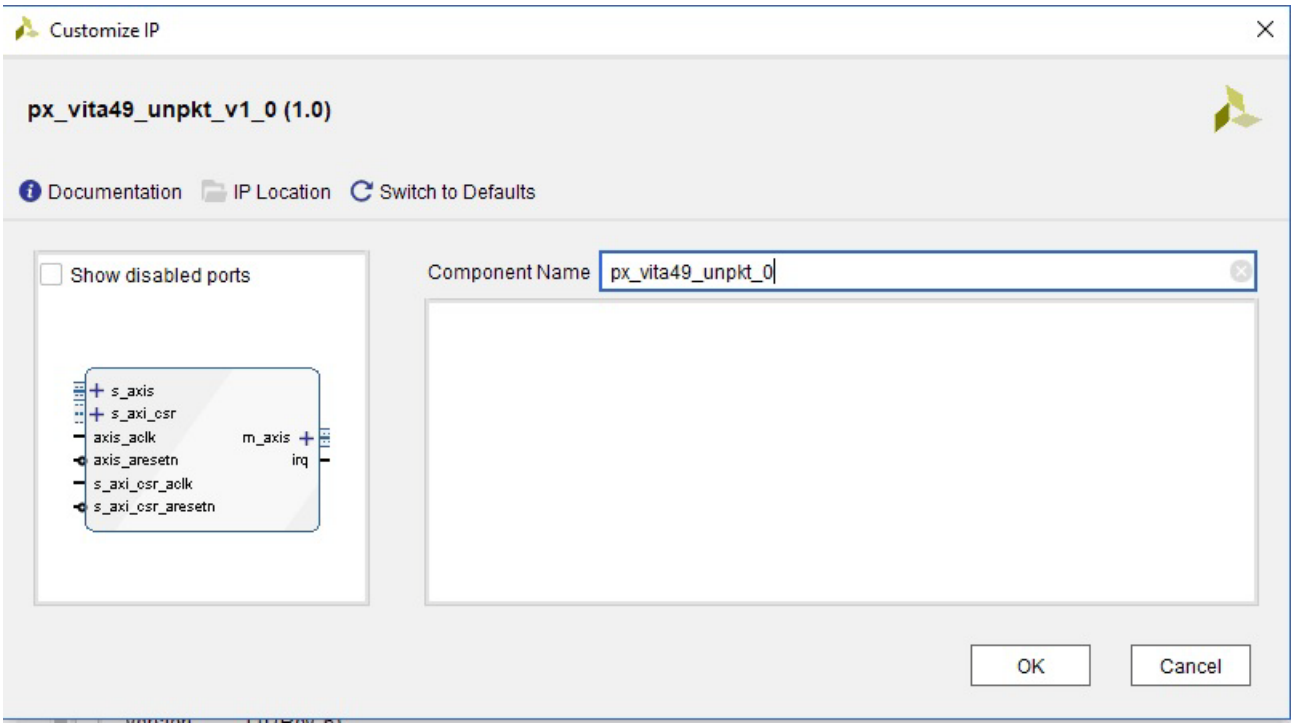
Figure 6–1: AXI4–Stream VITA 49 Unpacketizer Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_vita49_unpkt_1_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: AXI4–Stream VITA 49 Unpacketizer Core IP Symbol



6.2 User Parameters

This section is not applicable to this IP core.

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 VITA 49 Unpacketizer Core in Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the VITA 49 Unpacketizer Core. The neces−sary 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.

Clock Frequencies

The maximum **axis_aclk** frequency for this IP core is 200 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

Figure 6–3: AXI4–Stream VITA 49 Packet Header in Simulation Output with All Optional Fields Enabled

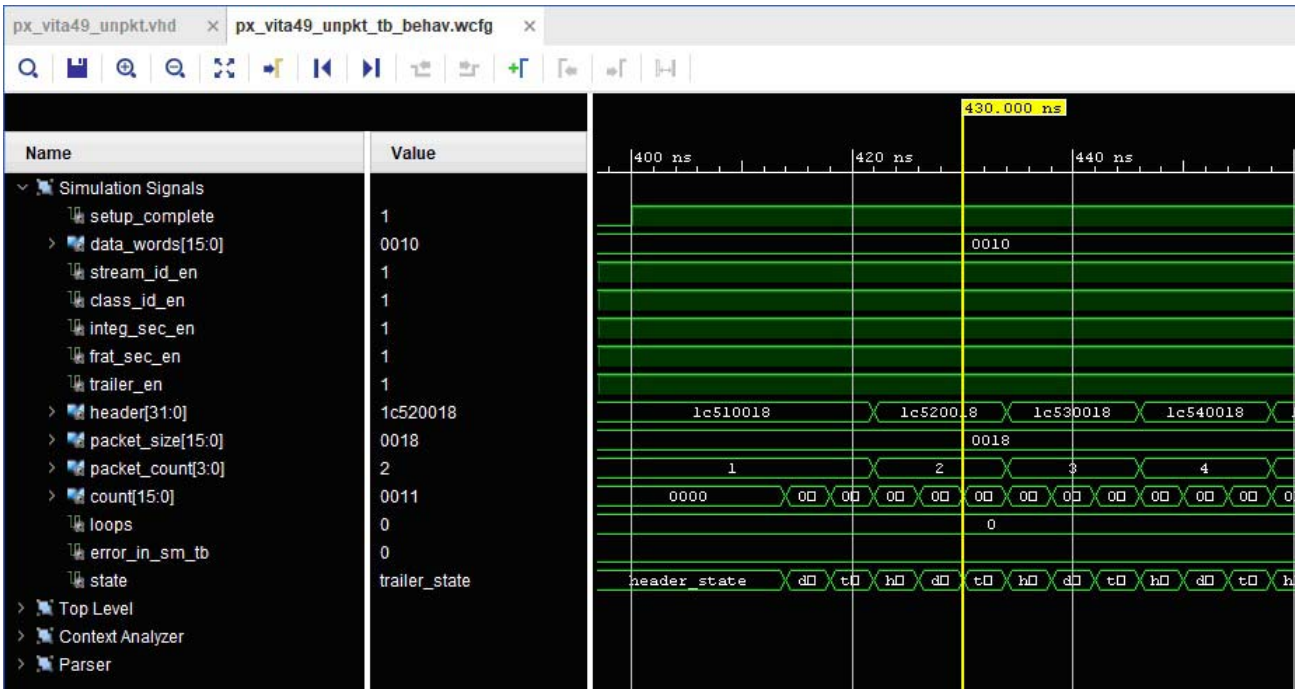


Figure 6–4: AXI4–Stream VITA 49 Unpacketizer Core Input Stream and Resulting Output Stream

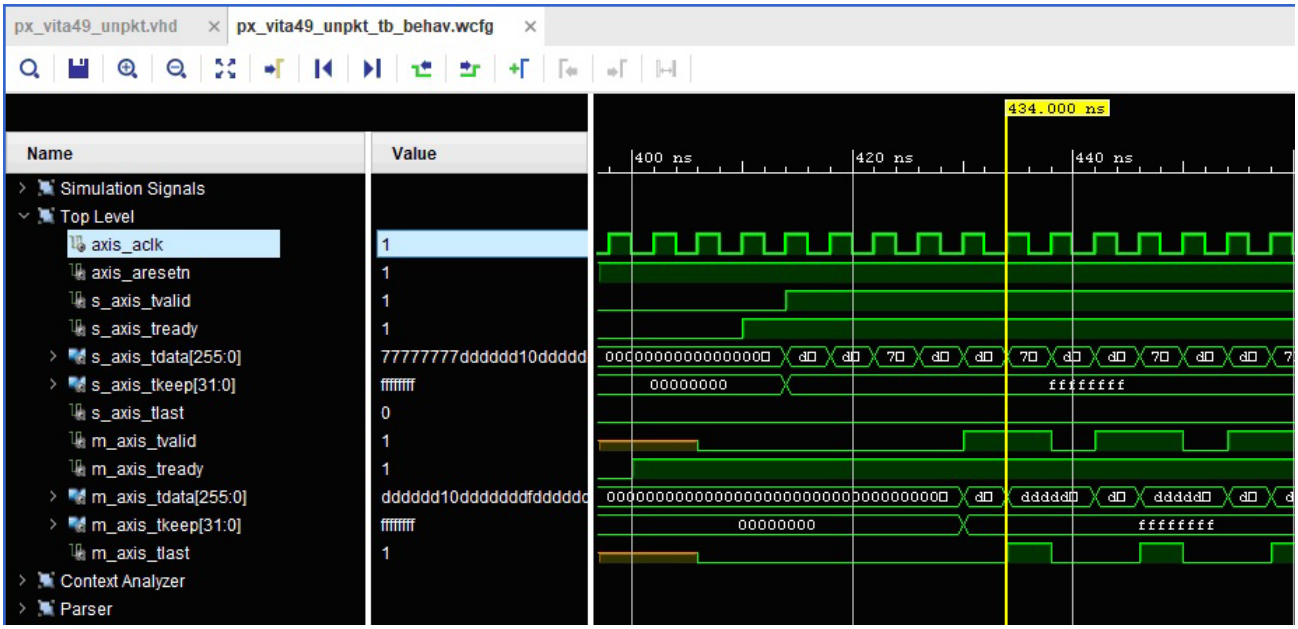


Figure 6–5: Context Analyzer Deciphering Header for Present Optional Fields and Packed Size

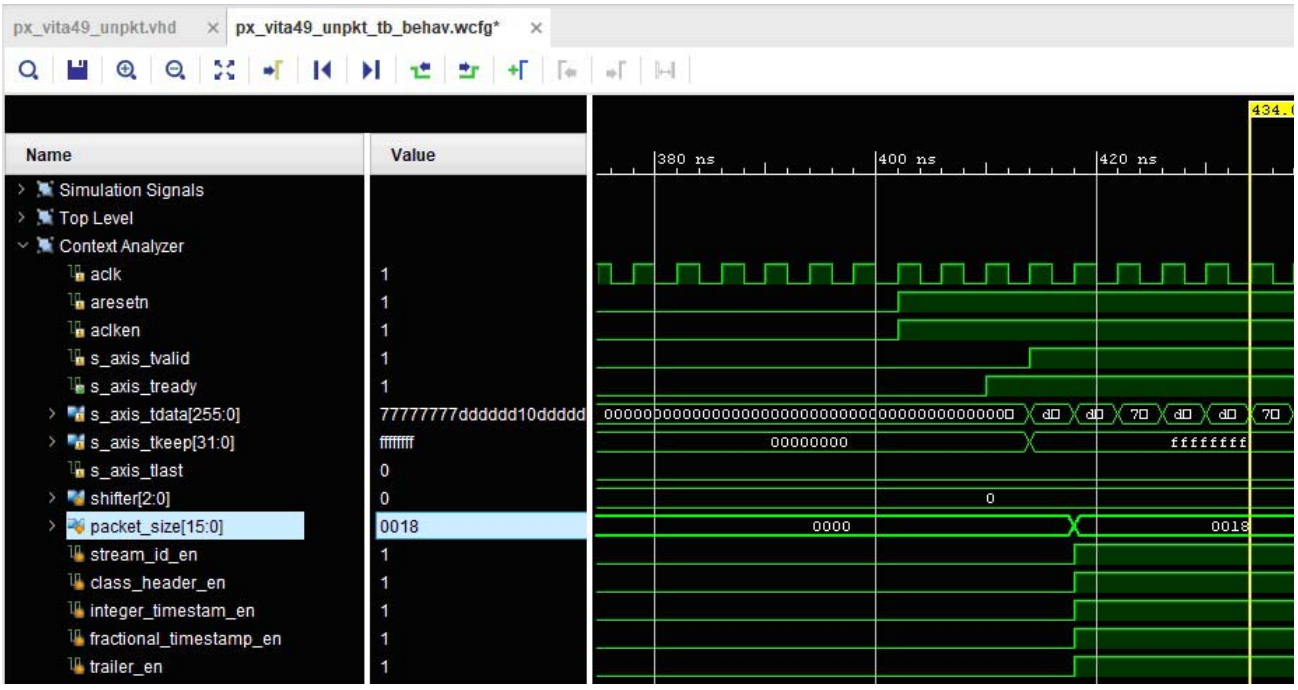


Figure 6–6: Parser Labeling VITA 49 Header and Optional Fields from Data. Data Fields are Output to `m_axis_tdata`.

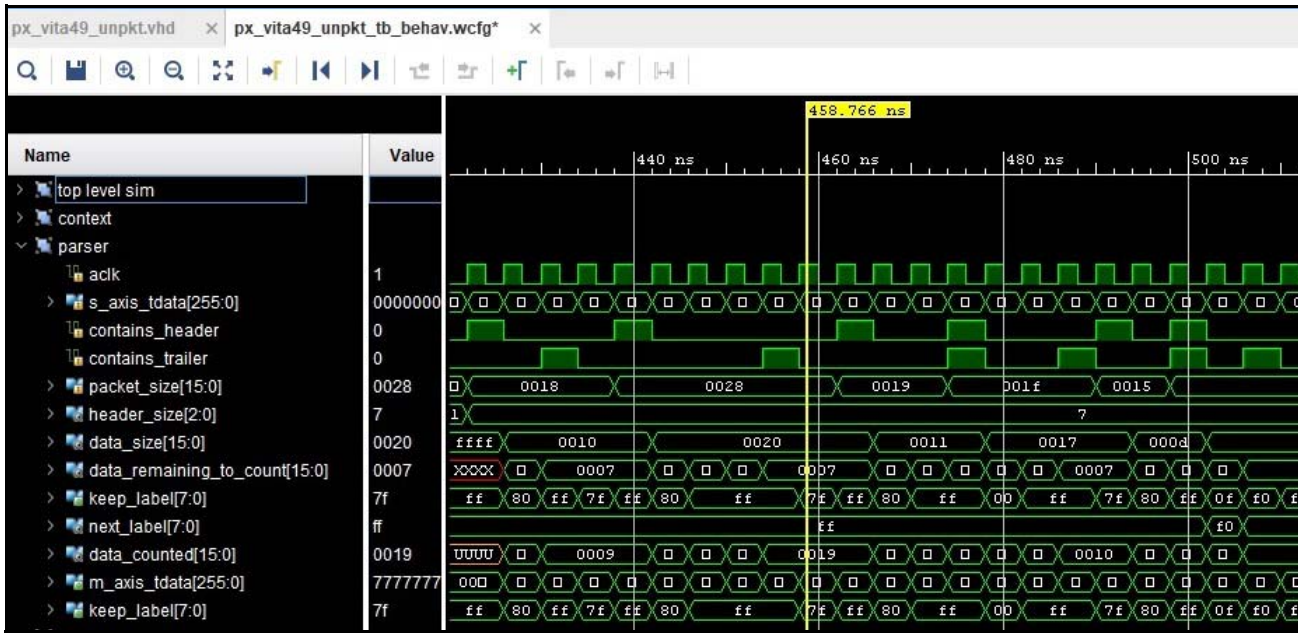
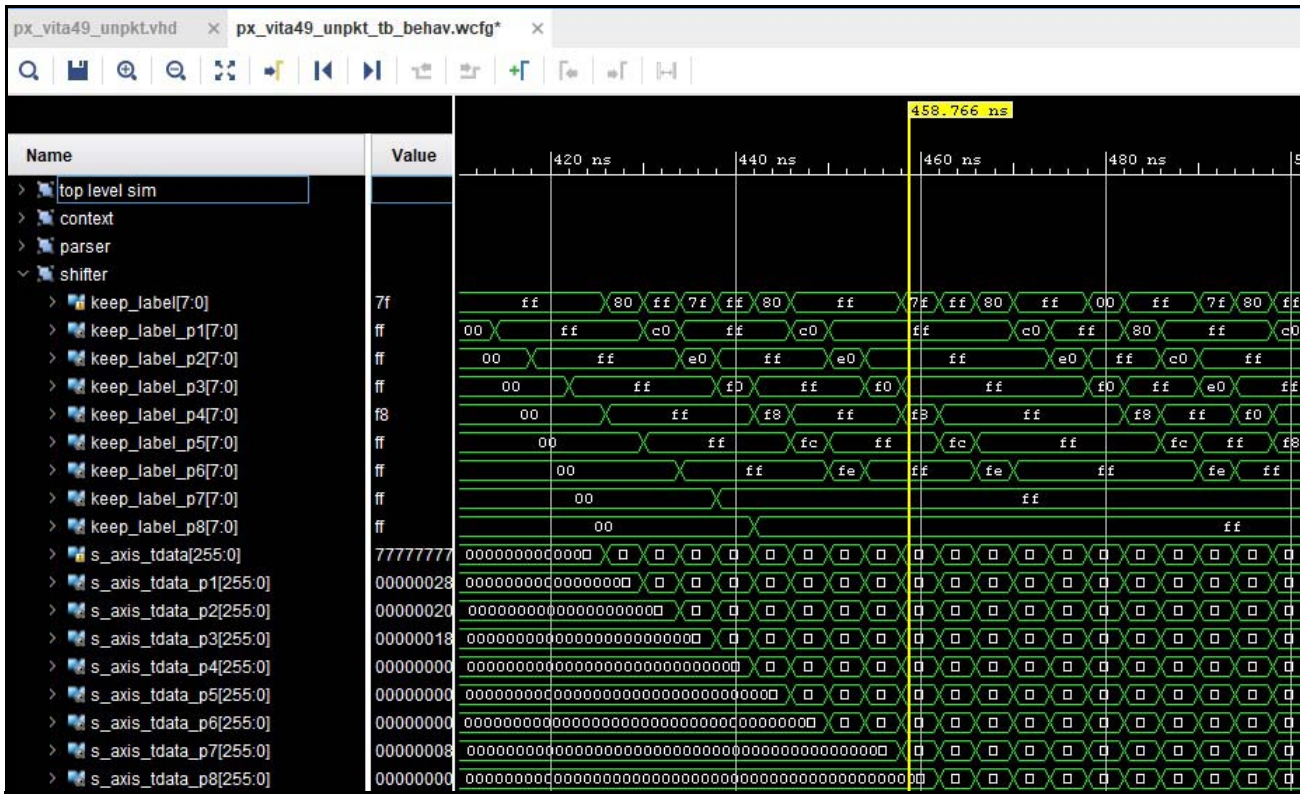


Figure 6–7: Shifter Shifting Out Labels Marked as '0.' An 8–pipeline Delay Accounts for Worst Case. All 256 bits Contain Header Info.



6.6 Synthesis and Implementation

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