

## **PHY IP Design Flow with Interlaken for Stratix V Devices**

AN-634-1.0

Application Note

This application note describes implementing and simulating the protocol-specific PHY intellectual property (IP) core in Stratix® V devices using the Interlaken PHY IP interface. You can use the reference design file described in this application note to evaluate the design flow of the PHY IP interface and the Stratix V device family.

The PHY IP interface, starting with the Quartus® software version 10.1, automatically configures the setting parameters in Stratix V devices for the physical coding sublayer (PCS) module, leaving a small number of parameters in the physical media attachment (PMA) module for your control.

The PHY IP design is modular and uses standard interfaces. All PHY IP modules include an Avalon<sup>®</sup> Memory-Mapped Management Interface to access control and status registers and an Avalon Streaming interface to connect to the MAC block for data transfer. Most of the PCS and PMA functions are implemented as hard logic to save FPGA resources. The PCS and PMA blocks cannot be separated or bypassed. The key interfaces are serial data from external interfaces, parallel data for internal PLD interface, and an internal control interface.

The PHY IP interfaces include the following:

- Avalon-ST TX/RX Interface—The PLD parallel interface for the transmit (TX) and receive (RX) data stream from the data packet of the TX and RX path of the PCS.
- Avalon-ST Ready Interface—Indication signals that the RX and TX paths are active and ready for data reception or transmission.
- Serial RX and TX Interface—The transceiver's input and output serial data.
- Avalon Memory-Mapped (Avalon-MM) Management Interface—Indirect addressing to access the PHY IP's internal registers and settings. The PHY IP expects phy\_mgmt\_address[8:0] as a word address, but because the byte address is the standard interface to a master, the Avalon master takes the byte address (mgmt\_address[9:0]) and converts it to a word address for the PHY IP.
- Clock Interface—The clock interface includes the user clock output for the RX and TX. Other inputs are the phase-locked loop (PLL), TX and RX clock, and PHY IP management clock.





San Jose, CA 95134 www.altera.com

101 Innovation Drive

© 2010 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX are Reg. U.S. Pat. & Tm. Off. and/or trademarks of Altera Corporation in the U.S. and other countries. All other trademarks and service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.









## Implementing an Interlaken Interface with the PHY IP

The Interlaken interface protocol allows the design of a narrow high-speed chip-to-chip interface for networking applications. The Interlaken interface's many advantages over XAUI and SPI 4.2 interfaces include low I/O count, flow control, low overhead framing, and extensive integrity checking. Interlaken is scalable and based on the SPI 4.2 structure to support 10-, 40-, and 100-Gbps systems. In an Interlaken configuration, Stratix V devices support a transmission data rate of 3.125, 5.0, 6.25, 6.375, or 10.3125 Gbps per lane.

### **Reference Design Flow for the Interlaken PHY IP example**

This section describes the procedures to run the Interlaken PHY IP reference design with a Stratix V device.

Use the Quartus II software version 10.1 or beyond to implement and compile your design. Currently there is no hardware platform; you can only complete a reference design simulation.

The following PHY IP creation shows how to use the MegaWizard<sup>™</sup> Plug-In Manager to generate the PHY IP (the PHY IP has already been created as part of the reference design).

Figure 2 shows the reference design directory structure.

#### Figure 2. Reference Design Directory Structure



### **Quartus II Software Top-Level Creation**

The PHY IP implementation makes the transceiver's protocol-specific interface a simple task when compared with using the ALTGX megafunction, where there are many variables to set for the transceiver.

To create a new Quartus II project file and a top-level design file, follow these steps:

- 1. Name the top-level verilog file top\_interlaken\_phy.v.
- 2. Launch the MegaWizard Plug-In Manager to create a new custom megafunction.
- 3. Select the **Interface Plug-Ins** drop-down menu and click **Interlaken PHY v10.1**. Select the output as verilog file **interlaken\_ip**.
- 4. In the **Interlaken PHY** dialog box, you can select the datapath mode, lane rate, and number of lanes. In this example, select **duplex mode** for the receive and transmit with a **5 Gbps** data rate for one lane, as shown in Figure 3.
- 5. Set the assigned meta frame length to **100 word**.
- 6. Select the parameters interface signals. The extra signals are word\_lock, sync\_lock, and CRC32 error and are part of the rx\_parrellel\_data bus. An tx\_coreclkin input port is created for driving the write side of the TX FIFO and an rx\_coreclkin input port is created for driving the write side of the RX FIFO.
- 7. Click Finish to create the PHY IP.

Figure 3 shows the Interlaken PHY IP settings.

#### Figure 3. Interlaken PHY IP

	at_interlaken_pcs			entation
Block Diagram				
	interla	ken phy		
clock clock clock conduit avalon_streaming conduit avalon_streaming conduit conduit	pil_ref_cik bc_corecikin tw_corecikin tw_ready tw_serial_data tw_paralle_data nc_ready mc_serial_data nc_dataout_bp nc_fifo_cir phy_mgmt_cik	b_cikout n_cikout b_datain_bp n_parallel_data	■ clock ■ clock ■ avalon_st ■ avalon_st	reaming
General Device family:	Stratix V	L	]	
Lane rate:	5000 Mbps			
Number of lanes:	1			
Meta frame length in word	\$ 100			
Parameters				
Extra signals     Create tx_coreclkin p     Create rx_coreclkin p	ort ort			
Info: interlaken_phy: 1	The input clock frequen	cy should be 250.000000 N	Hz	

The top-level verilog design consists of synthesizable code with an Avalon master, pseudo-random binary sequence (PRBS) generator and checker, and a frequency checker. All the necessary design files are included in the reference design package. All the modules were created as part of this reference design. Use an Avalon master for each PHY IP core you use in your design. The top-level design instantiates the following modules:

- *Interlaken\_phy*.v (generated by the MegaWizard Plug-In Manager)
- Custom\_avalon\_master.v
- Prbs\_generator.v
- Prbs\_checker.v
- Frequency checker.v

Table 1 lists the important status and error signals.

Table 1. Status and Error Signals (Part 1 of 2)

Status and Error Signals	Description
rx_sync_lock	The $sync_{lock}$ status signal indicates when four sync words are detected.
rx_word_lock	The word_lock status signal indicates when the first alignment pattern is detected.

Status and Error Signals	Description
rx_framing_error	Changes to "1" when a framing error is detected.
rx_crc32_error	Changes to "1" when an CRC32 error is detected.
rx_scrambler_error	Changes to "1" when a scrambler error is detected.
rx_sync_word_error	Changes to "1" when a missing_sync error is detected.
txclkfreq_error	Changes to "1" when the $tx_clkout$ frequency is not within range.
rxclkfreq_error	Changes to "1" when the ${\tt rx\_clkout}$ frequency is not within range.
data_error	Changes to "1" when the PRBS checker detects an error.
verifier_lock	Changes to "1" when the verifier detects a lock.

Table 1. Status and Error Signals (Part 2 of 2)

### **Custom Avalon Master**

The Quartus II software version 10.1 and before does not allow the SOPC builder or Qsys to create any Avalon interface instance. Therefore, Altera designed a custom Avalon master, called *custom\_avalon\_master.v*, to accept the byte address and to provide the word address to the PHY IP.

The Avalon master accepts the byte address mgmt\_address[9:0] and delivers the word address mgmt\_address[8:0] to the PHY IP. The custom Avalon master was designed for an Interlaken interface, but you can use the same master for other PHY IP because they all have a similar structure. In this example, the internal serial loopback feature, which exercises the PMA's control and status register (CSR), is enabled.

The custom Avalon master is the heart of the interface between the FPGA logic core and the PHY IP. You can use this module as a template to modify and adapt to other PHY IP protocols. Each PHY IP protocol type will have a different set of internal registers to read or write through the Avalon master. The input and output data bus width between the PHY IP and the logic core is 32 bits.

Figure 4 shows the custom\_avalon\_master signal flow, which was generated into a symbol.

Figure 4. Custom Avalon Master Signal Flow (custom\_avalon\_master)

		Parameter	Value	Туре
		phy_serial_loopback_addr	0110000100	Unsigned Binary
		log_ch_addr	1000000000	Unsigned Binary
		pcs_status_addr	1000000100	Unsigned Binary
stom_avalon_master				
phy_mgmt_olk	phy_mgmt_writedata[310]			
phy_mgmt_reset	phy_mgmt_write	1		
tx_ready	phy_mgmt_address[80]			
rx_ready	phy_mgmt_read			
phy_mgmt_readdata[310]	rx_syne_lock			
phy_mgmt_waitrequest	rx_word_lock			
	nx_framing_error			
	nx_orc32_error			
	rx_scrambler_error			
	rx_sync_word_error			

Even though this reference design is for the Interlaken protocol, you can reuse some of the key building blocks for other PHY IP protocols. The code you can reuse in the reference design is the PRBS generator, PRBS checker, and, most importantly, the Avalon master module, as shown below in Figure 5.





### **PRBS Generator and Checker**

A 64-bit PRB23 data generator and check modules are used for loopback testing.

### **Top-Level Testbench**

This section describes the register transfer level (RTL) simulation testbench and the simulation results of the reference design.

The RTL simulation testbench provides you with the RTL design verification at the software level. The reference design provides a simple testbench for the loopback environment between the TX and the RX function using a PRBS as a test pattern source. The testbench checks for the status and error signals and displays the required output information.

The device under test (DUT) is the PHY IP. A clock generator was created in the testbench for the PHY IP's clock interfaces, which includes the transceiver reference clock (pll\_ref\_clk) and the data path clock (phy\_mgmt\_clk).

### **Run the ModelSim Example Script**

To run a simulation of the reference design in the ModelSim<sup>®</sup>-Altera software, follow these steps:

- 1. Start the ModelSim Starter Edition version 6.6c or beyond.
- 2. Under the File menu, click **Change Directory** and change to the location of the reference design that you have installed.

Ensure the *modelsim\_example\_script.tcl* file is located the directory folder.

3. At the ModelSim command prompt, type do modelsim\_example\_script.tcl to run the simulation shown in Figure 6.

### Figure 6. ModelSim Script

```
Transcript =
# Reading C:/Altera/10.1/modelsim_ase/tcl/vsim/pref.tcl
cd {C:/Documents and Settings/tamdo/Desktop/Interlaken/Ref design/alt_interlaken_phy}
ModelSim> do modelsim_example_script.tcl
```

The **.tcl** script file is an example file for compilation and simulation of the Interlaken PHY instance in ModelSim. All the necessary verilog HDL or VHDL library files are included with the reference design project so that you can pick either language. You can modify this script file and use it to compile and simulate your design depending on new requirements.

Use the top-level testbench file *top\_interlaken\_phy\_tb.v* to complete the simulation. This file instantiates the top-level PHY IP project under the test and clock generator. The testbench checks for the status and error signals and displays any error information.

### Simulation Results

A successful simulation has the following key status signals:

- TX PCS is ready for this example at 56661 ps
- RX PCS is ready for this example at 156651 ps
- rx\_sync\_lock at 7055961 ps
- rx\_word\_lock at 7102623 ps
- Verifier lock at 7740000 ps

Figure 7 shows the ModelSim simulation's output at the end of a successful simulation.

#### Figure 7. Successful ModelSim Simulation Results



You can further select the signals that you want to see from the ModelSim Objects window (Figure 8).

To select different signals to display in the Wave window (Figure 9), follow these steps:

- 1. Highlight the signal that you want to display in the Wave window (Figure 8).
- 2. On the Add menu, select To wave and Signals in design (Figure 9).
- 3. On the VSIM command prompt, type run 1000 to start the waveform window.

### Figure 8. ModelSim's Objects Window

Layout Simulate	120	2 0- 4 8	는 막 파 🕹 🖽	A OTO DAY	and a	ALL A DESCRIPTION OF	
Layout Simulate				0.9 33	01	🖛 🔿 👔 🎼 10	0 ps 🛊 1 1 1
a set of the set of the set of the set of the	- Col	lumnLayout Def	ault		-	N B B B	1 10 2 2 3
	Г						
🕅 sim		Harx 🔽 🔿	ojects :				10
Instance 🛆 Desig	n unit Des	tign ⊾ _ ▼Nam	ie .	Value		Kind	Mode
😤 #vsim_capacity#	For	eign 🚽 🚄	phy_mamt_ck_reset			Register	Internal
altera_Insim_functi altera	a_insi ViPa	ackar, 🖌 🚄	dk150	5t0		Net	Internal
get_real_value altera	a_Insi Fun	ictior 🚽	dk125	5t0		Net	Internal
get_time_string alter.	a_Insi Fun	iction 🗳	dk250	St1		Mat	Internal
get_time_value alter	a_Insi Fun	iction 🖌 🚄	dk156	5t0		Net	Internal
🖃 🗾 std 🦳 🛛 std	VIPa	acka; 🖌 🍊	rx_sync_lock	Sti		Net	Internal
- 📶 mailbox 🛛 std	SVP	<sup>t</sup> aran 🧳	rx_word_lock	St1		Net	Internal
🔄 🗾 process 🛛 std	SVC	Jass 🖌 🍊	rx_framing_error	5t0		Net	Internal
💿 🗾 semaphore 🛛 std	SVC	Jass 🖌 🍊	rx_crc32_error	St0		Net	Internal
<pre>sv_generic_csr_h sv_g</pre>	eneric VIPa	ackar 🖌 🍊	rx_scrambler_error	5t0		Net	Internal
top_interlaken_ph top_i	nterla Mod	dule 🛛 💫	rx_sync_word_error	5t0		Net	Internal
- #ALWAYS#85 top_i	nterla Pro	cess 🚽	txckfreq_error	510		Net	Internal
-> #ALWAYS#98 top_i	nterla Pro	cess 🗸	rxclifreq_error	St0		Net	Internal
-> #ALWAYS#110 top_	nterla Pro	cess	reg_out_file	2		Integer	Internal
<ul> <li>A state of the second se</li></ul>	stade the	and a state		and the second se		and the second se	
#ALWAY5#123 top_i	riceria Prov	cess	data error	StU		Net	Internal

Figure 9. Waveform Window



# **Document Revision History**

Table 2 lists the revision history for this application note.

### Table 2. Document Revision History

Date	Version	Changes
December 2010	1.0	Initial release.