Intel® Arria® 10 10GBASE-KR PHY IP Design Example User Guide

Intel Quartus Prime Pro Version: **22.3**

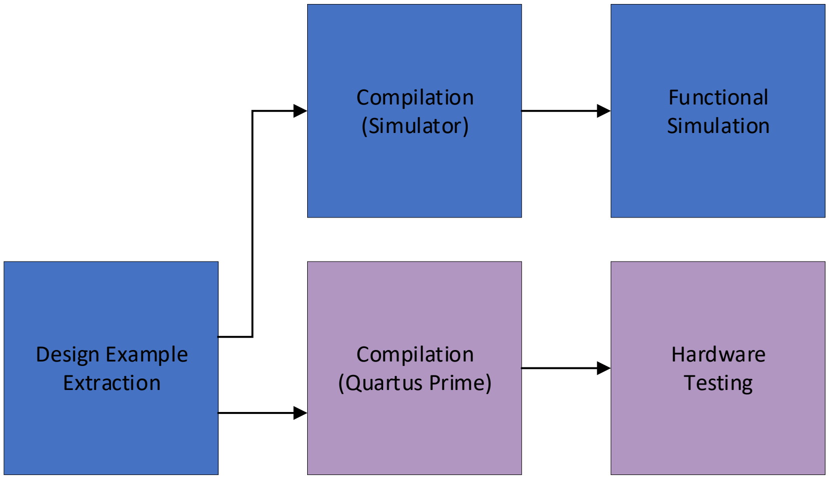
Last updated: **July 31, 2023**

Contents

1.0 Quick Start Guide

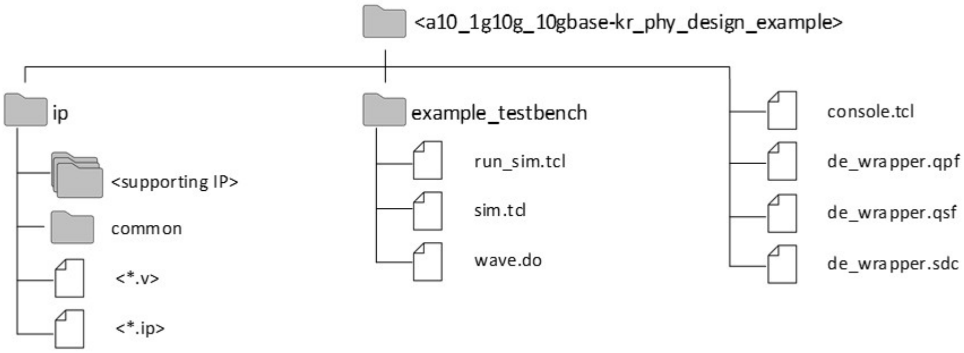
The Intel® Arria® 10 10GBASE-KR PHY IP design example provides a simulation testbench and a hardware design example that supports simulation and hardware testing. You can download the compiled hardware design and run it on the Intel® Arria® 10 GX Transceiver Signal Integrity Development Kit.

Figure 1-1 Development Stages for the Design Example



1.1 Design Example Directory Structure

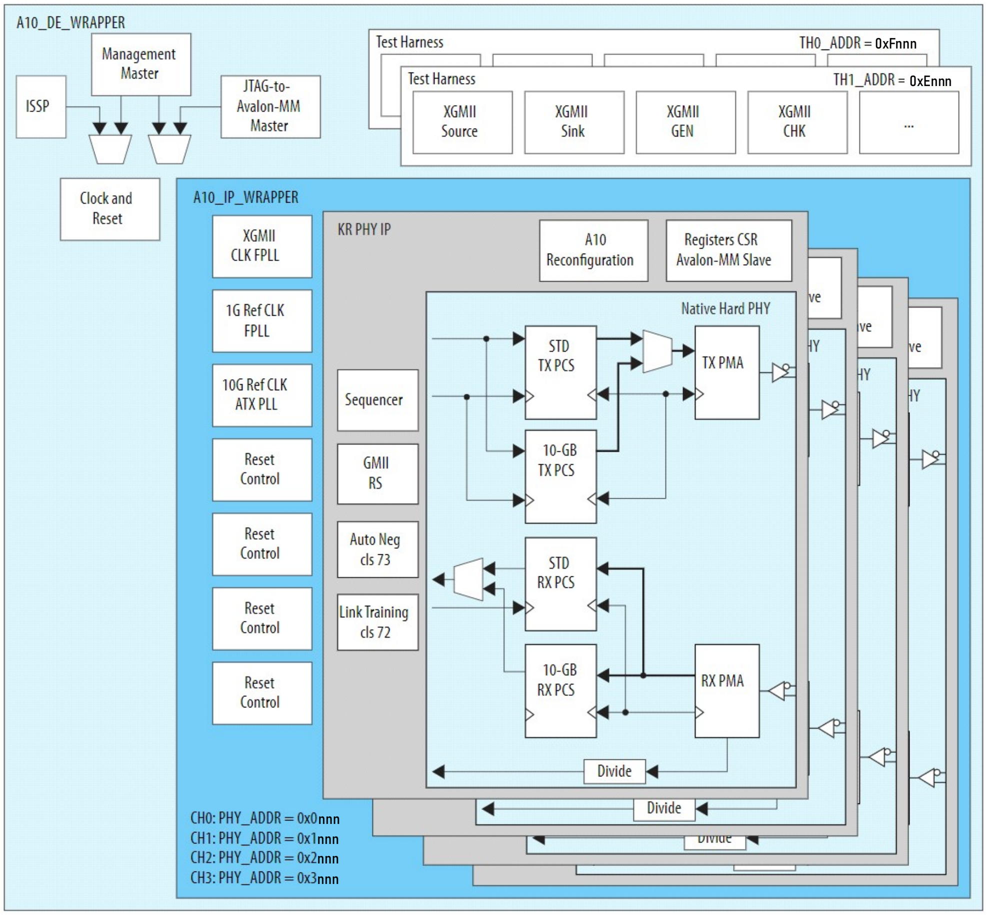
Figure 1-2 Intel® Arria® 10 10GBASE-KR Design Example Directory Structure



The hardware configuration and test files are in `<design_example_dir>`. The simulation files are in `<design_example_dir>/example_testbench`. The IP and RTL files are in `<design_example_dir>/ip`.

1.2 Hardware Design Example Components

Figure 1-3 Intel® Arria® 10 10GBASE-KR Design Example High Level Block Diagram



The Intel® Arria® 10 10GBASE-KR hardware design example includes the following components:

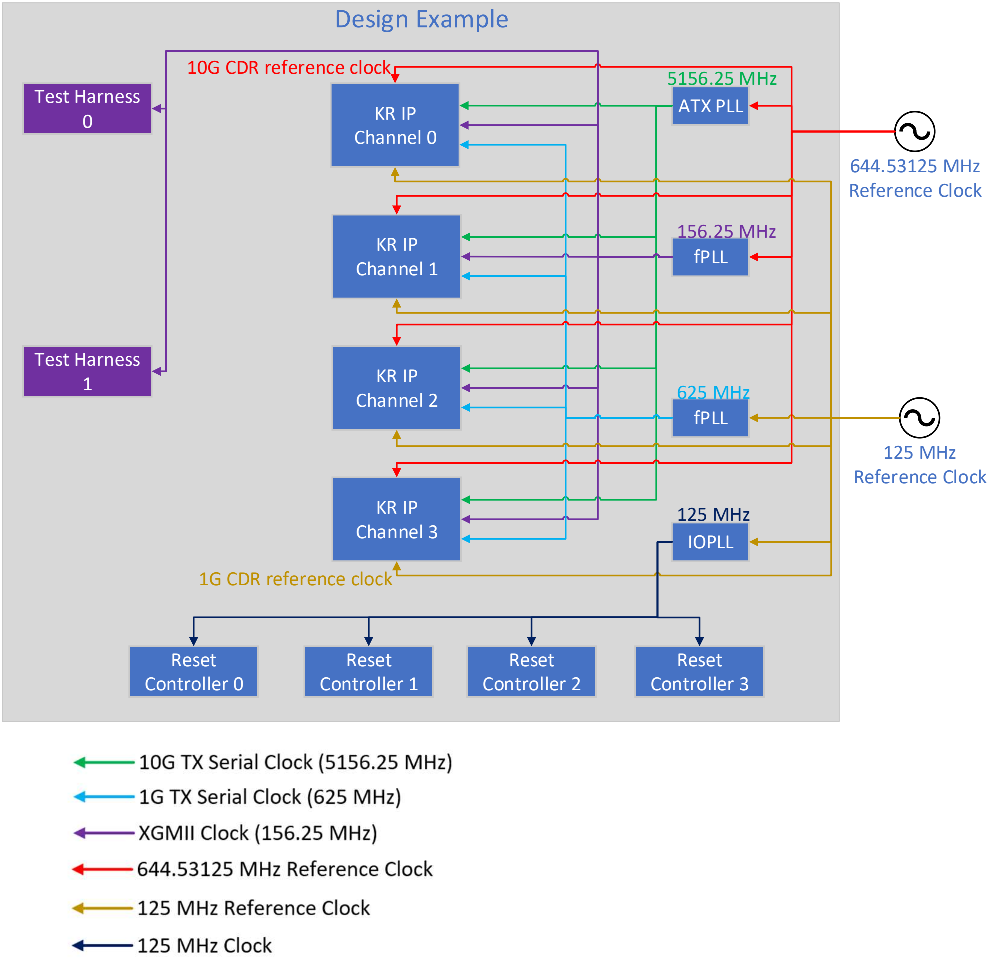
- 1G/10GbE and 10GBASE-KR PHY Intel® Arria® 10 FPGA IP
- Transceiver PHY Reset Controller Intel FPGA IP.
- ATX PLL to generate the high-speed serial clock to drive the device transceiver channel (10G mode).
- fPLL to generate the high-speed serial clock to drive the device transceiver channel (1G mode).
- fPLL to generate XGMII clock.
- IOPLL to generate a 125 MHz clock.
- Packet Generator and Packet Checker (Test Harness)

- JTAG controller that communicates with System Console. You communicate with the client logic through the System Console

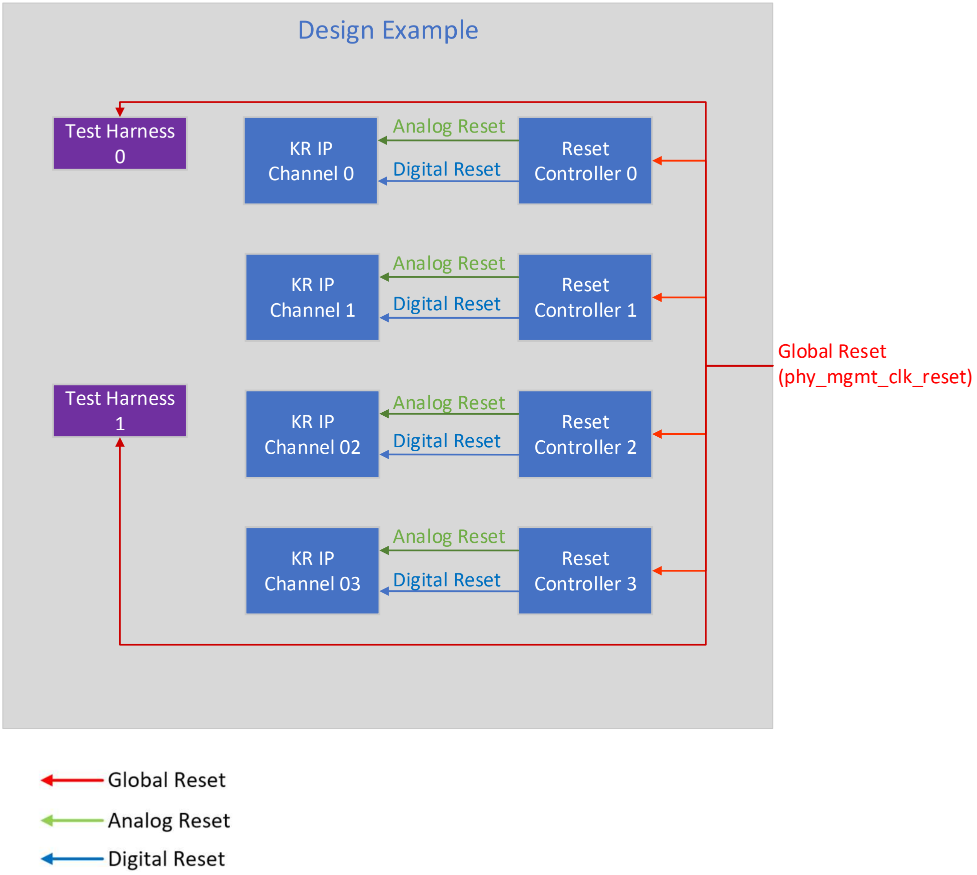
Table 1-1 Intel® Arria® 10 10GBASE-KR Hardware Design Example File Descriptions

File Name	Description
de_wrapper.qpf	Intel® Quartus® Prime project file
de_wrapper.qsf	Intel® Quartus® Prime project settings file
de_wrapper.sdc	Synopsys Design Constraints file. You can copy and modify this file for your own design.
console.tcl	Main file for accessing System Console

1.2.1 Clocking Scheme



1.2.2 Reset Scheme



1.3 Simulation Design Example Components

Table 1-2 Intel® Arria® 10 10GBASE-KR Testbench File Descriptions

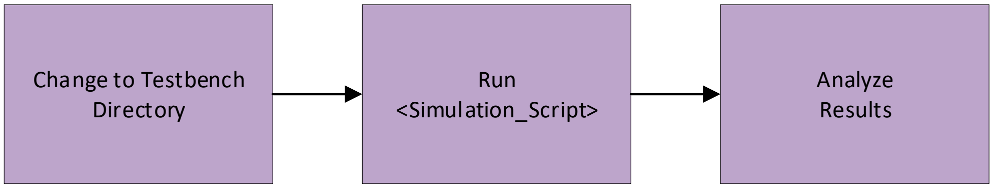
File Name	Description
run_sim.tcl	Top-level Siemens QuestaSim* script to run the testbench.
sim.tcl	Script to compile and simulate the simulation design example.
wave.do	Simulation waveform file.

The simulation design example top-level test file is `de_wrapper.sv` located in `<design_example_dir>/ip`.

1.4 Simulating the Intel® Arria® 10 10GBASE-KR Design Example Testbench

You can compile and simulate the design by running a simulation script from the command prompt.

Figure 1-4 Simulation Procedure



Follow these steps to simulate the testbench:

1. Navigate to the testbench simulation directory `<design_example_dir>/example_testbench`
2. Run the simulation script for the supported simulator of your choice. The script compiles and runs the testbench in the simulator. Refer to Table 1-3 "Steps to Simulate the Testbench".

Table 1-3 Steps to Simulate the Testbench

Simulator	Instruction

Modelsim* or QuestaSim*	In the command line, type:
	`vsim -do run_sim.tcl`
	If you prefer to simulate without bringing up the GUI, type:
	`vsim -c -do run_sim.tcl`

3. Analyze the results. The successful testbench sends and receives XGMII blocks and displays “Test case Passed”.

The following sample output illustrates a successful simulation test run:

```
#
# Start frame detected, byteslip 0, time 28622807
# Start frame detected, byteslip 0, time 131630987
# Start frame detected, byteslip 0, time 604405883
#
# Info: Management_Master_bfm_done = 1; mgmt program done
at time          632062000000
#
# Test case Passed
#
```

1.5 Compiling and Configuring the Design Example in Hardware

To compile the hardware design example and configure it on your Intel device, follow these steps:

1. In the Intel® Quartus® Prime Pro Edition software, open the Intel® Quartus® Prime project `<design_example_dir>/hardware_test_design/de_wrapper.qpf`.

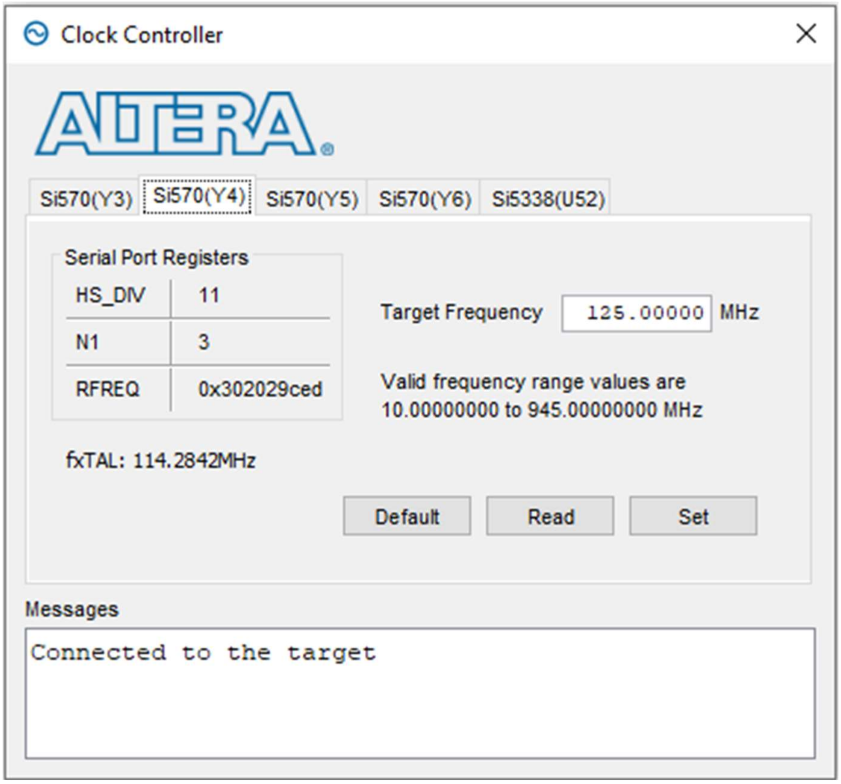
2. On the Processing menu, click **Start Compilation**.

3. After successful compilation, a .sof file is available in `<design_example_directory>/hardware_test_design/output_files`.

   - Connect the Intel® Arria® 10 GX Transceiver Signal Integrity Development Kit to the host computer.

   - Launch the Clock Controller application, which is part of the development kit installation package. This can be downloaded from the Intel® Arria® 10 GX FPGA Development Kit page on the Intel website here. The clock controller application is located in <package_installation_directory>/examples/board_test_system/ClockController.exe

Figure 1-4 Clock Controller



Serial Port Registers

HS_DIV	11
N1	3
RFREQ	0x302029ced

Target Frequency 125.00000 MHz

Valid frequency range values are 10.00000000 to 945.00000000 MHz

fXTAL: 114.2842MHz

Messages

```
Connected to the target
```

Set new frequency for the design example as following:

- Si570(Y4): Set to the value of `Target Frequency` as 125MHz

4. On the **Tools** menu, click **Programmer**.

5. In the **Programmer**, click **Hardware Setup**.

6. Select a programming device.

7. Select and add the Intel® Arria® 10 Transceiver Signal Integrity Development Kit to your Intel® Quartus® Prime session.

8. Ensure that **Mode** is set to **JTAG**.

9. Select the Intel® Arria® 10 device and click **Add Device**. The Programmer displays a block diagram of the connections between the devices on your board.

10. In the row with your `.sof`, check the **Program/Configure** box for the `.sof` file.

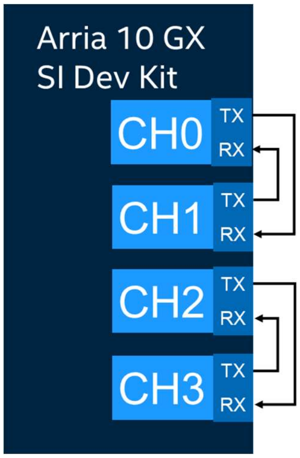
11. Click **Start**.

1.6 Testing the Hardware Design Example

After you compile the Intel® Arria® 10 10GBASE-KR PHY IP core design example and configure it on your Intel® Arria® 10 device, you can use the **System Console** to program the IP core.

Follow these steps to test the hardware design example on the **System Console**:

1. Cross connect the differential pairs of channels 0 and 1, and channels 2 and 3 using 2.4mm SMA cables, as described below. For testing of Auto Negotiation and Link Training, it is implemented in channels 2 and 3. Skip this step for internal loopback mode testing of channels 0 and 1.



Default pin connections on Intel® Arria® 10 GX Transceiver Signal Integrity Development Kit are as follows.

Channel 0 and 1 pins:

- Connect GXBL_1G_RX0p to GXBL_1G_TX1p
- Connect GXBL_1G_RX0n to GXBL_1G_TX1n
- Connect GXBL_1G_RX1p to GXBL_1G_TX0p
- Connect GXBL_1G_RX1n to GXBL_1G_TX0n

Channel 2 and 3 pins:

- Connect GXBL_1G_RX3p to GXBL_1G_TX4p
- Connect GXBL_1G_RX3n to GXBL_1G_TX4n
- Connect GXBL_1G_RX4p to GXBL_1G_TX3p
- Connect GXBL_1G_RX4n to GXBL_1G_TX3n

2. After the hardware design example is configured on the Intel® Arria® 10 device, in the Intel® Quartus® Prime Pro Edition software, on the **Tools** menu, click **System Debugging Tools > System Console**.
3. Type `source console.tcl` to open a connection to the JTAG master.

You can program the IP core with the following design example commands:

- `loop_on`: Turns on internal serial loopback.
- `loop_off`: Turns off internal serial loopback.
- `reconfig_read` *`<channel>`* *`<addr>`* : Returns the IP core register value at <*channel*> and <*addr*>.
- `reconfig_write` *`<channel>`* *`<addr>`* *`<data>`* : Writes <*data*> to the IP core register at <*channel*> and <*addr*>.

- `rst` *`<channel>`* : Reset the instance of KR IP.
- `dis_max_wait_timer`: Disables the link training max wait timer.
- `dis_nonce`: Ignores nonce during AN. This allows AN to work with internal loopback mode.
- `rd_seq_stat`: Display status from sequencer block.
- `rd_an_stat`: Display status back from AN block.
- `rd_lt_stat`: Display status back from LT block.
- `start_xgmii_th<`*`channel`*`>`: Starts sending 10G packets from test harness. Only works for channel 0 and channel 2.
- `rd_pkt_cnt_th<`*`channel`*`>`: Read non idle data sent during the test. Expected value is 418. Only works for channel 0 and channel 2.

The following sample output illustrates a successful hardware test run:

```
% rd_seq_stat
Channel:0  Address 0x4B0 == 0x00020102
           Address 0x4B1 == 0x00000401
Channel:1  Address 0x4B0 == 0x00020102
           Address 0x4B1 == 0x00000401
Channel:2  Address 0x4B0 == 0x00020102
           Address 0x4B1 == 0x00000401
Channel:3  Address 0x4B0 == 0x00020102
           Address 0x4B1 == 0x00000401



% rd_an_stat
Channel:0 Address 0x4C0 == 0x00000000
          Address 0x4C2 == 0x00000000
Channel:1 Address 0x4C0 == 0x00000000
          Address 0x4C2 == 0x00000000
Channel:2 Address 0x4C0 == 0x00000001
          Address 0x4C2 == 0x000040f4
Channel:3 Address 0x4C0 == 0x00000001
          Address 0x4C2 == 0x000040f4

% rd_lt_stat
Channel:0 Address 0x4D0 == 0x00000000
          Address 0x4D2 == 0x00000000
Channel:1 Address 0x4D0 == 0x00000000
          Address 0x4D2 == 0x00000000
Channel:2 Address 0x4D0 == 0xe1145111
          Address 0x4D2 == 0x00000001
Channel:3 Address 0x4D0 == 0xe1145111
          Address 0x4D2 == 0x00000001

% start_xgmii_th2

% rd_pkt_cnt_th2
Non-idle count for Channel-2 is :- 0x000001a2
Non-idle count for Channel-3 is :- 0x000001a2
```

2.0 Design Example Description

This design example demonstrates the basic functions of Backplane-KR variant of 1G/10GbE and 10GBASE-KR PHY Intel® Arria® 10 FPGA IP core. It consists of 4 instances of 10GBASE-KR PHY, ATX PLL, fPLLs, Transceiver PHY Reset Controller, JTAG to Avalon Master Bridge and PHY-level data transmit/receive modules as shown in Figure 1-3 Intel® Arria® 10 10GBASE-KR Design Example High Level Block Diagram.

The following IP parameter settings were used to generate this design example:

Table 2-1 IP Parameters of Channel 0 and 1 10BASE-KR PHY IP

IP Parameter	Value
General Options	
IP variant	Backplane-KR
Initial datapath	10G
Enable IEEE 1588 Precision Time Protocol	Disabled
Reference clock frequency	644.53125 MHz
Include FEC sublayer	Disabled
Enable 1Gb Ethernet protocol	Enabled
Enable Auto-Negotiation	Disabled
Enable Link Training	Disabled

Table 2-2 IP Parameters of Channel 2 and 3 10BASE-KR PHY IP

IP Parameter	Value
General Options	
IP variant	Backplane-KR
Initial datapath	10G
Enable IEEE 1588 Precision Time Protocol	Disabled
Reference clock frequency	644.53125 MHz
Include FEC sublayer	Disabled
Enable 1Gb Ethernet protocol	Enabled

Enable Auto-Negotiation	Enabled
Enable Link Training	Enabled

2.1 Hardware and Software Requirements

To test the example design, use the following hardware and software:

- Intel® Quartus® Prime Pro Edition software
- Intel® Arria® 10 GX Transceiver Signal Integrity Development Kit
- System Console
- ModelSim*, QuestaSim* simulator
- Optional: SMA cables

2.2 Design Example Behavior

The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

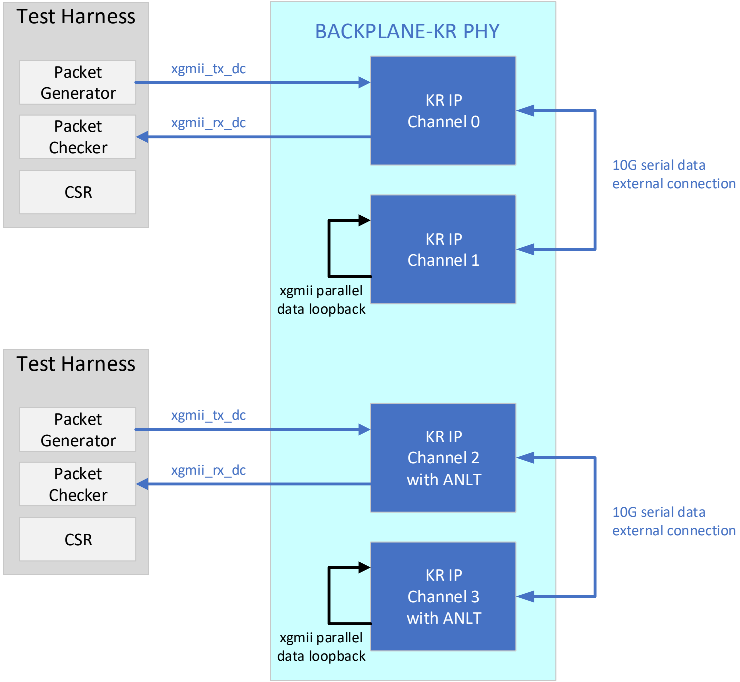
For the hardware design example, only channels 0 and 1 can be run using internal loopback mode. If loopback mode is off for those channels, then external connections are required. Channels 2 and 3 must have external connections. These external connections are described in section 1.6.

The traffic between channels is generated and checked by a test harness. This is illustrated in figure 2-1. Channels 0 and 1 runs the backplane-KR variant IP without Auto Negotiation and Link Training. Channels 2 and 3 demonstrates the 10GBASE-KR with Auto Negotiation and Link Training (ANLT) enabled.

For 10GBASE-KR with ANLT, using the start_xgmii_th2 command in system console sends out test packets from the test harness through the channel 2 transmitter. The test packets are received by the receiver in channel 3. The received xgmii parallel data then goes through a loopback internally and the data is sent back out on the channel 3 transmitter to the channel 2 receiver. The rd_pkt_cnt_th2 command can be used in system console to read the number of packets received which is 418 for every start_xgmii_th2 command.

Similarly, the respective commands can be used to test out channel 0 and 1 for the backplane-KR variant IP without ANLT.

Figure 2-1 Packet generation and checker for Backplane-KR IP



2.3 Design Example Interface Signals

Table 2-3 Intel® Arria® 10 10GBASE-KR Hardware Design Example Interface Signals

Signal	Direction	Description
pll_ref_clk_10g	Input	644.53125 MHz 10G mode PHY reference clock.
pll_ref_clk_1g	Input	125MHz 1G mode PHY reference clock.
clk125	Input	125MHz system clock.
usr_seq_reset_pb3	Input	User push button (PB3) that resets the sequencer of Channel 0 and 1 Transceiver PHY. Active low.
usr_seq_reset_pb2	Input	User push button (PB2) that resets the sequencer of Channel 2 and 3 Transceiver PHY. Active low.
ch0_rx_serial_data	Input	Channel 0 Transceiver PHY input serial data.
ch1_rx_serial_data	Input	Channel 1 Transceiver PHY input serial data.
ch2_rx_serial_data	Input	Channel 2 Transceiver PHY input serial data.
ch3_rx_serial_data	Input	Channel 3 Transceiver PHY input serial data.
ch0_tx_serial_data	Output	Channel 0 Transceiver PHY output serial data.
ch1_tx_serial_data	Output	Channel 1 Transceiver PHY output serial data.
ch2_tx_serial_data	Output	Channel 2 Transceiver PHY output serial data.
ch3_tx_serial_data	Output	Channel 3 Transceiver PHY output serial data.
ch0_rx_recovered_clk	Output	Channel 0 RX recovered clock.
ch2_rx_recovered_clk	Output	Channel 0 RX recovered clock.
user_led0	Output	System clock status signal connected to on-board LED.

user_led2	Output	Reference clock status signal connected to on-board LED.
user_led4	Output	XGMII clock status signal connected to on-board LED.

2.4 Intel® Arria® 10 10GBASE-KR Design Example Registers

Registers of this design example can be accessed via Avalon Memory-mapped interface.

Table 2-4 Intel® Arria® 10 10GBASE-KR Design Example Register Map

Offset	Name	Description
0x0000 - 0x04FF	CH0_PHY	Channel 0 PHY access
0x1000 - 0x14FF	CH1_PHY	Channel 1 PHY access
0x2000 - 0x24FF	CH2_PHY	Channel 2 PHY access
0x3000 - 0x34FF	CH3_PHY	Channel 3 PHY access
0xE000 – 0xEFFF	Test Harness 2	**Refer to** Table 2-5 Test Harness Register Map
0xF000 – 0xFFFF	Test Harness 0	**Refer to** Table 2-5 Test Harness Register Map

Related Information

Intel® Arria® 10 10GBASE-KR PHY Registers

Table 2-5 Test Harness Register Map

Offset	Bit	R/W	Name	Description
0x00	0	RWSC	Start XGMII packets	When set to 1, sends the specified number of XGMII packets. This bit will be cleared automatically.
	8	RWSC	Start GMII/ GIGE packets	When set to 1, sends the specified number of GMII packets. This bit will be cleared automatically.
0x01	0	RW	Reset Hold	When set to 1, holds the channel in reset. This bit must be written to 0 for normal operation.
	8	RW	Generator Reset	When set to 1, holds the XGMII and GMII generators in reset. Bit must be written to 0 for normal operation.

	16	RW	Enable 10G Reverse Loopback	When set to 1, enables the 10G reverse parallel loopback. i.e., the received data from xgmii_rx_dc input is sent back on the xgmii_tx_dc output.
0x02	0	R	XGMII rx_ready	When asserted, indicates the block synchronizer has successfully established synchronization. The incoming XGMII data block locks the receiver.
	1	R	XGMII checker_pass	When asserted indicates that the received frames are exactly same as the transmitted frames. When XGMII rx_mismatch asserted, this is set to 0.
	2	R	XGMII rx_mismatch	Asserted even if single received frame doesn't match the transmitted frame.
	3	R	XGMII fifo_full	Shows the XGMII FIFO on the TX side is full. When set to 1, indicates error condition.
	4	R	XGMII test_done	When asserted indicates that the transmitter has sent all frames. Each frame is 64-bit wide data and 2-bit control.
	5	R	XGMII frame_done	When asserted indicates that the transmitter finished sending frames and then goes low automatically.
	6	R	XGMII test_pass	When asserted, XGMII has received all sent packets and there is no mismatch in received data. XGMII status= rx_data_ready & checker_pass & ! rx_mismatch
0x03	0	R	GMII rx_ready_gmii	When asserted, indicates the word aligner has successfully established synchronization. The incoming GMII data block locks the receiver.
	1	R	GMII led_an	When asserted, indicates completion of auto-negotiation.

	2	R	GMII packet_complete_gen	When asserted indicates that the packet transmission is completed.
	3	R	GMII packet_complete_chk	When asserted indicates that the packet check is completed.
	4	R	GMII rx_mismatch_gmii	Asserted even if single received frame doesn't match the transmitted frame.
	5	R	GMII ch_pass	When asserted, GMII has received all sent packets and there is no mismatch in received data. GMII status= packet_complete_gen & packet_complete_chk & ! rx_mismatch_gmii.
0x04	19:0	RC	CH0 XGMII idle error count	Count of the number of non-idle 66-bit words received. Reset to 0 on register read.
0x05	19:0	RC	CH1 XGMII idle error count	Count of the number of non-idle 66-bit words received. Reset to 0 on register read.
0x06 to 0xFF	Reserved			