Arria10 GX ‘SuperLite II V3’ Demo Design V16.1.2 for the Arria10 GX SI Board
4 Lanes @ 10.3125 Gbps routed to the QSFP+ Module

Peter Schepers

Technology Specialist FAE High Speed Interfaces

April 14th, 2017

Introduction & Motivation

This design illustrates how to use the Superlite II V3 concept on Arria10GX

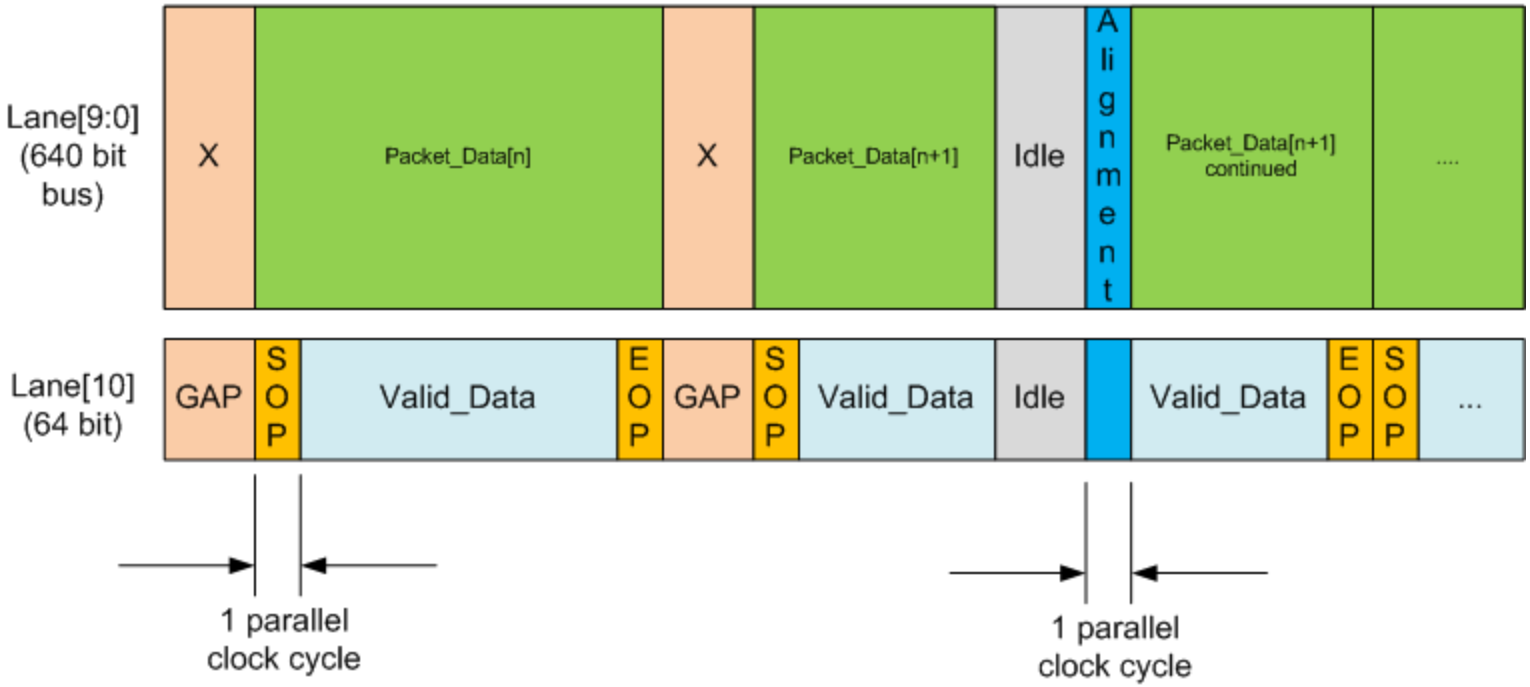
The V3 version of Superlite II is based on the V2 version. The V2 version uses a valid signal in combination with the transmit data and there is also a ready signal provided which can potentially halt the transmit traffic for a short interval (based on filling level of Transmit FIFO). This interface is equivalent to the Avalon Streaming Interface.

The V3 version adds additional functionality :

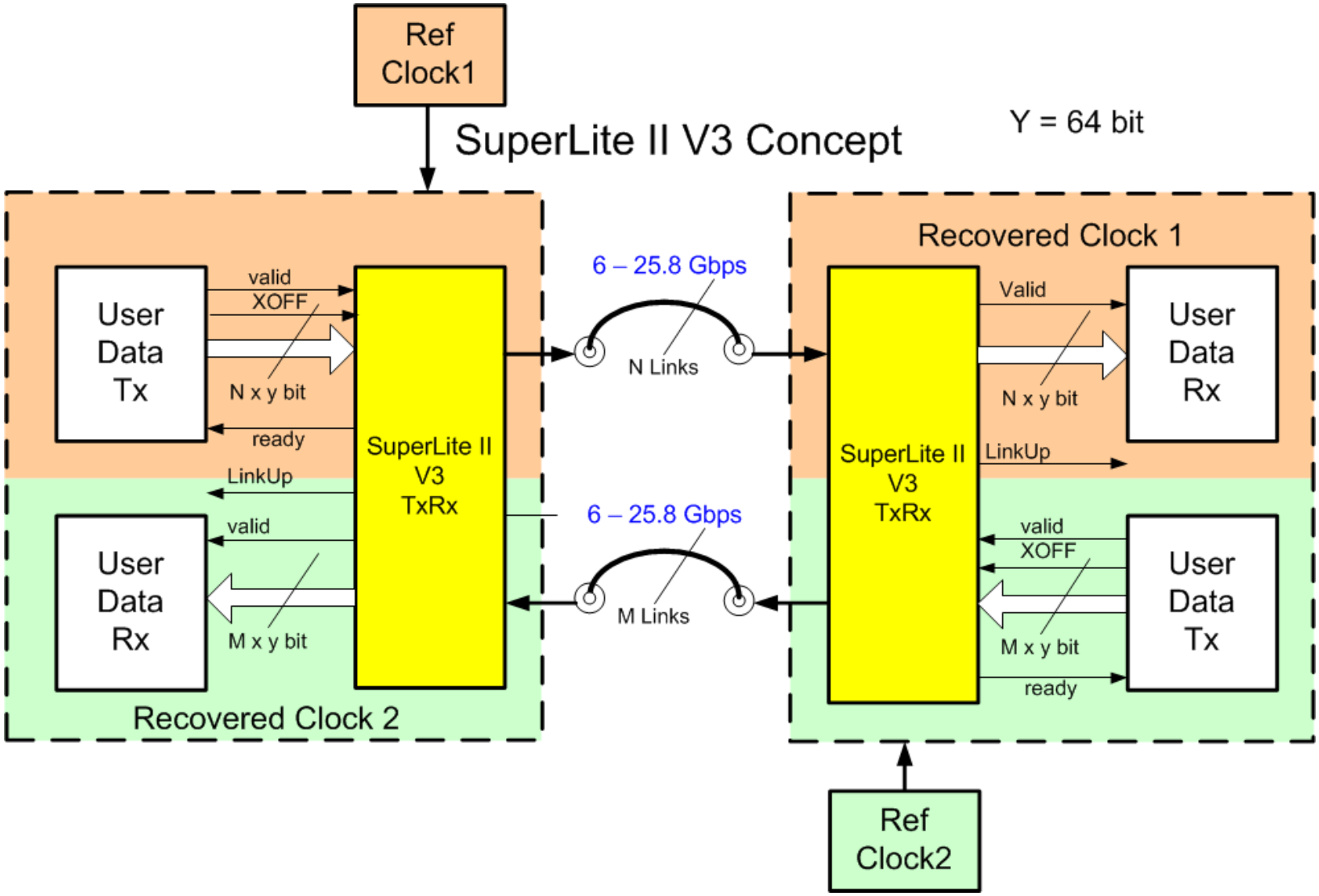
- Link information is exchanged between the local and remote side and results in a LinkUp when both local and remote side are aligned.
- XOFF is used in order to allow for backpressure or Flow Control: ask remote side to stop sending traffic (when local side e.g. is not capable of receiving the data).
- Because of the exchanges between local and remote side the protocol is now a fully bidirectional protocol (V1 and V2 versions could be used fully unidirectional)

How to use Superlite II V3 in Packet mode ?

- By nature the Superlite II V3 is using streaming data (Avalon streaming based) so not framed or packetized . One could however support also a framed mode (or packet mode) by using one of the lanes to transport the packet related information (SOP, EOP, etc) . This does require an additional lane to provide the packet overhead information. This concept has been succesfully implemented on Stratix V GX (with 11 lanes) and can easily be repeated on Arria10 (with higher rates and including KR-FEC).



‘SuperLite II V3’ Concept



‘SuperLite II V3 with KR-FEC’ Concept

Transport data from point A to point B as simple as possible multiple bidirectional serial links. Note that the number of lanes in each direction do not have to be equal.

V3 version of the protocol exchanges information between local and return side (handshaking) as well as XON/XOFF control (backpressure)

On the transmit side data will only be written in the FIFO when the valid signal is asserted, the fifo also can issue a backpressure signal (indicated by the ‘ready’ signal to indicate if data can be transmitted. This is equivalent to the Avalon Streaming interface.

When XOFF is received from the remote side, all traffic will be halted on the local side (link remains up though).

On the receive side it will be based on the recovered clock with data valid. The combination of data valid with the recovered clock is frequency locked to the clock used at the transmit side.

Uses 64/66 Encoding in combination with scrambling.

Multiple lanes are bonded, there is no real limit on the number of lanes that can be used, apart from potential limitations to compile very large number of lanes. Typically 4 or 10 lanes will be used. Also a single lane is possible.

For word alignment and lane alignment idle characters and alignment characters will be sent at regular intervals based on the filling level of the Fifo at the transmit side.

User data bandwidth per lane must be less then (Lane_rate*64)/66 Gbps in order to leave bandwidth for idle and alignment characters. The factor of 64/66 is the additional overhead due to 64b66b encoding on the lane. The Superlite II TX V3 module will automatically pause Tx user data if needed to insert idle and alignment characters. In cases where the Tx user data is not constantly being driven then there will not be any backpressure effectively.

‘Superlite V3’ Concept (continued)

Parameters for backpressure and idle time can be configured in RTL through a compile time parameter (could also be made programmable).

User data is always 64 bit per lane because of the 64b66b encoding.

There are basically 3 ways to clock the transmit data for the V3 version.

- First option is to use a totally independent clock for clocking the transmit data. If you want to make sure that all data is being sent without being halted, the clock rate should be lower than the tx_clkout * Ratio (with Ratio = (READ_LENGTH - IDLE_LENGTH))/(READ_LENGTH). In the demo design READ_LENGTH is 0xFF and IDLE_LENGTH is 0x07, but this can be configured to other values in RTL (you could change the design to do this in software as well).
- Second option is to take the reference clock (provided the referenceclock is datarate/66). In this case the transmit data will be periodically halted in order for the transmit logic to insert idle and alignment characters at regular intervals (again defined by READ_LENGTH and IDLE_LENGTH)
- 3rd option is to use the 156.25 Mhz clock generated by the transceiver (datarate/66), which is the method used in this design.

In case there is not always transmit data to be sent you could end up with a mode of operation where the ready willl always be high.

In case you want to sent across data at all times, you should use Superlite instead (the V1 version).

‘SuperLite II V3 4 lane Demo Design

Goals :

- Demonstrate the Superlite II V3 concept on Arria10 GX
  - Use Arria10 GX Signal Integrity board as platform for demonstration.
    - Uses the 4 channels routed to the QSFP+.
    - Using 2 separate boards is also possible if connection through optical QSFP+ module is done.
  - Line rate in this demo is 10.3125 Gbps per lane (because of the QSFP+ module) but could be go all the way up to 25.781 Gbps.
  - 4 lanes will be used, therefore aggregate line rate is 40 Gbps.
  - User data width is 256-bits (64 bits per lane)
  - Actual throughput depends on the configured “read time” and “idle time” parameters. In the demo the “read time” is 255 clock cycles and “idle time” is 7 clock cycles but these can be easily changed. The “read time” would normally be set at a higher value like e.g. 66536 clock cycles in order to increase the efficiency.
  - The user data is generated based on the 156.25 Mhz clock generated by the Native PHY (data rate divided by 66). As it receives backpressures from the Superlite II V3 Tx module the data will be halted at specific times based on the Fifo Filling level.
  - To create the 256 bit user data a combination of different counterpatterns and prbspatterns are used across the 4 lanes.

Goals (continued)

Goals :

- Use Reconfiguration Management Interface for Native PHY and LC PLL through QSYS system.
- Through ADME Transceiver Toolkit support is automatically available.
- Use Nios II processor as controller/monitor
- Allows to perform ODI Measurement from within the Nios command shell, measuring both 2D eye as well as bathtub curve on the selected lane and provide statistics (Eye height, width etc).
- The PMA settings of the lanes can dynamically be modified
- Includes Testbench

‘SuperLiteII V3’ Resources for 4 Lanes

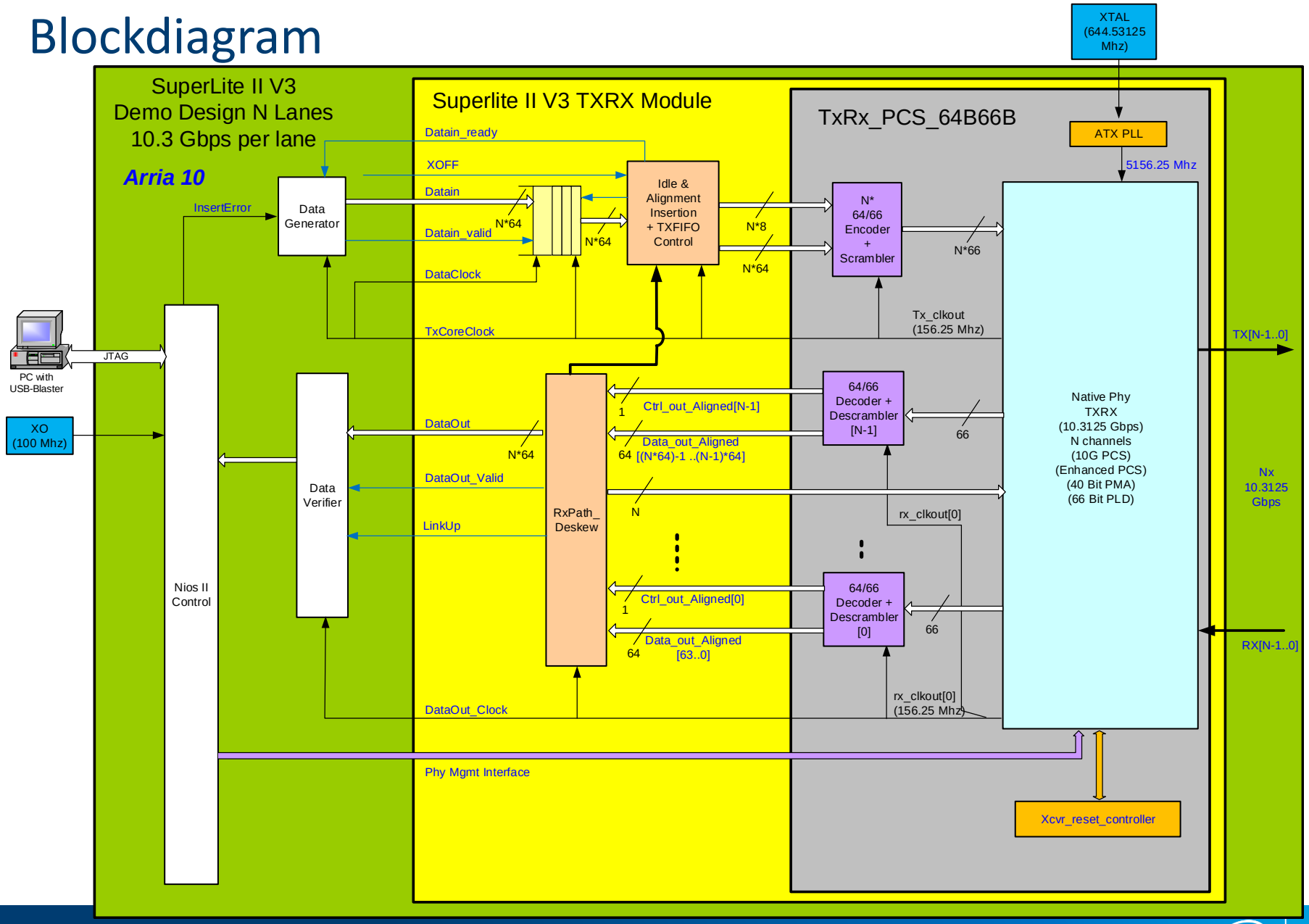
SuperliteII V3 TxRx Module Resources

Note : The largest % of the resources is taken by the additional logic enabled in the Native PHY for debug purposes (like ADME etc) This is additional debug logic that is not required for a full design). The numbers below have substracted the logic resource for the transceivers.

- ~ 1800 ALUT’s
- ~ 1040 ALM’s
- ~ 2000 Dedicated logic registers
- 7 M20K’s
- 8K Block Memory Bits

Entity:Instance	ALMs needed [=A-B+C]	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	M20Ks	DSP Blocks
superliteii_txrx_module:superliteii_txr...	2533.6 (3.0)	4009 (6)	4385 (5)	8192	7	0
Reset_Synchro:Reset_Synchro_...	1.0 (1.0)	0 (0)	4 (4)	0	0	0
Reset_Synchro:Reset_Synchro_...	1.0 (1.0)	0 (0)	4 (4)	0	0	0
Rx_Path_Deskew:Rx_Path_inst	98.0 (98.0)	136 (136)	46 (46)	0	0	0
Tx_Ratematcher_Control:Tx_Rat...	187.1 (153.2)	228 (178)	407 (306)	8192	7	0
txrx_pcs_64b66b:xcvr_txrx_inst	2243.5 (5.7)	3639 (9)	3919 (8)	0	0	0
Decoder_66b64b:\Generate...	85.5 (24.5)	166 (44)	154 (31)	0	0	0
Decoder_66b64b:\Generate...	85.3 (24.3)	166 (44)	153 (31)	0	0	0
Decoder_66b64b:\Generate...	85.0 (23.5)	166 (44)	153 (31)	0	0	0
Decoder_66b64b:\Generate...	85.5 (24.5)	166 (44)	157 (32)	0	0	0
Encoder_64b66b:\Generate_...	76.7 (15.7)	144 (22)	189 (67)	0	0	0
Encoder_64b66b:\Generate_...	77.3 (16.2)	144 (22)	189 (67)	0	0	0
Encoder_64b66b:\Generate_...	80.8 (19.6)	144 (22)	190 (67)	0	0	0
Encoder_64b66b:\Generate_...	80.0 (19.0)	144 (22)	189 (67)	0	0	0
xcvr_pll:xcvr_pll_inst	31.6 (0.0)	58 (0)	23 (0)	0	0	0
xcvr_reset_rx:xcvr_reset_r...	26.0 (0.0)	49 (0)	43 (0)	0	0	0
xcvr_reset_tx:xcvr_reset_tx...	31.3 (0.0)	59 (0)	55 (0)	0	0	0
xcvr_superlite_ii:xcvr_txrx_i...	1492.8 (0.0)	2224 (0)	2416 (0)	0	0	0

Blockdiagram



Tx Path

A datastream is generated at a clock of 156.25 Mhz @ 256 bit when DataIn_ready is high. The datapattern is a combination of 4 60-bit Prbs-23 (each with their own starting value) and one 16-bit counterpattern.

- The 16-bit counterpattern is spread across the 4 lanes, 4 bits per lane. This is to make sure any deskew on the lanes would be immediately spotted.

Since idle and alignment characters have to be inserted at regular intervals the data stream will be halted at regular intervals (based on the Fifo Filling level) in order to insert the required idle and alignment characters. This is indicated by the Superlite II TX module by the DataIn_ready signal.

In case XOFF is received from the Remote side the DataIn_Ready signal will be low to avoid any traffic being sent.

Single biterrors can be inserted using inserterror input. As there is one PRBSGenerator per lane, the biterror insert will produce 4 bit-errors at a time in fact.

This data is presented to the Superlite II Tx Module

The data is written into the TxFifo using the 156.25 Mhz as write clock and using the DataIn_Valid as write-enable.

The read clock for the TxFifo is the same 156.25 Mhz clock which is driven from the tx_clkout of the transceiver PHY

During the intervals that readenable is deactivated, idle characters and the alignment characters will be inserted in the datastream which is fed to the 64b66b encoder blocks.

Tx Path (continued)

The 64b66 encoding is very straightforward : bits 65..64 are set to “10” for control data and “01” for data (so neutral disparity)

In terms of idle character a value of “1C1C….1CBC” is being used. The alignment character is sent as the last control character. The lower 16 bits of this control word (at the alignment position) encodes the status of the link in addition to the alignment indication.

If no local alignment has been found : the last control word sent is “..7C7C”

If local wordalignment has been found : the last control word sent is “..FD7C”

If XOFF is being sent to the remote side : the last control word sent is “..5C7C”

If local side is not transmitting : the last control word sent is “..3C7C”

If local side is not transmitting and XOFF is being sent to the remote side : last control word sent is “…9C7C”.

Tx Path (continued)

In order to make sure there is DC balance and enough transitions present on the serial links a selfsynchronizing scrambler is used (the same as being used in 10GbE). The scrambler will scramble only the 64 databits (the framing bits on bits [65..64] are not scrambled since they are required for the framing. The polynomial used for the scrambler is 1 + x^39 + x^58 according to the IEEE.

The resulting 66-bit is sent to the Native PHY where the gearbox converts it from 66-bit to 40-bits PMA

Finally the data is serialized (using 40-bit serializer) and sent out as serial datastreams (10.3125 Gbps per lane).

Rx Path

After deserialization the data is presented to the KR-FEC Decoder inside the Native PHY where it will automatically align on the incoming Frames and decodes the FEC frames according to the IEEE802.3 FEC decoding.

As the FEC frames automatically on the incoming 66 bit signal no additional bitslip logic is required (frees up additional resources).

This is all happening on all lanes at the same time and everything is clocked based on the recovered clock of channel 0. The phasecompensation FIFOs inside the 10G PCS will be used to make sure lanes 1 to 4 have the right phaserelationship. Note that the recovered clock on lanes 1 to 4 are the same frequency as lane 0 but have a different phase relationship, hence the need for the phasecompensation FIFOs.

Rx Path (continued)

Based on the position of the alignment word, the delay for each lane is measured, the statemachine inside the Rx_path_deskew module will use the information of that measured delay to control a special Rx FIFO signal “rx_fifo_rd_en” on all RxFIFO’s to deskew all the lanes.

Once lane alignment is achieved the 256-bit combined data in addition with a data valid signal will be sent to the data verifier.

Here PRBS and counterverification will take place and single biterrors will be counted.

Prbslocked is asserted when 128 consecutive valid Prbs-23 60-bit words have been received on each of the lanes and de-asserted when 128 consecutive non-valid Prbs-23 60-bit words have been received.

Countlocked is asserted when 128 consecutive valid 16-bit counter words have been received on the parts of the lanes they were received.

There are 4 60 bit PRBS Verifiers (one for every lane) and 1 16-bit CountVerifiers to validate the 256 bit data.

Rx Path (continued)

The V3 version adds an additional Rx State machine to the logic to determine the local Rx state and it depends on the incoming “alignment” control word

If last control word received is “…FD7C” it means the remote side has reached Wordalignment, if local side also has reached wordalignment this results in Linkup to be asserted, indicating that both sides are up and running. This can be used as the trigger to sent data across e.g.

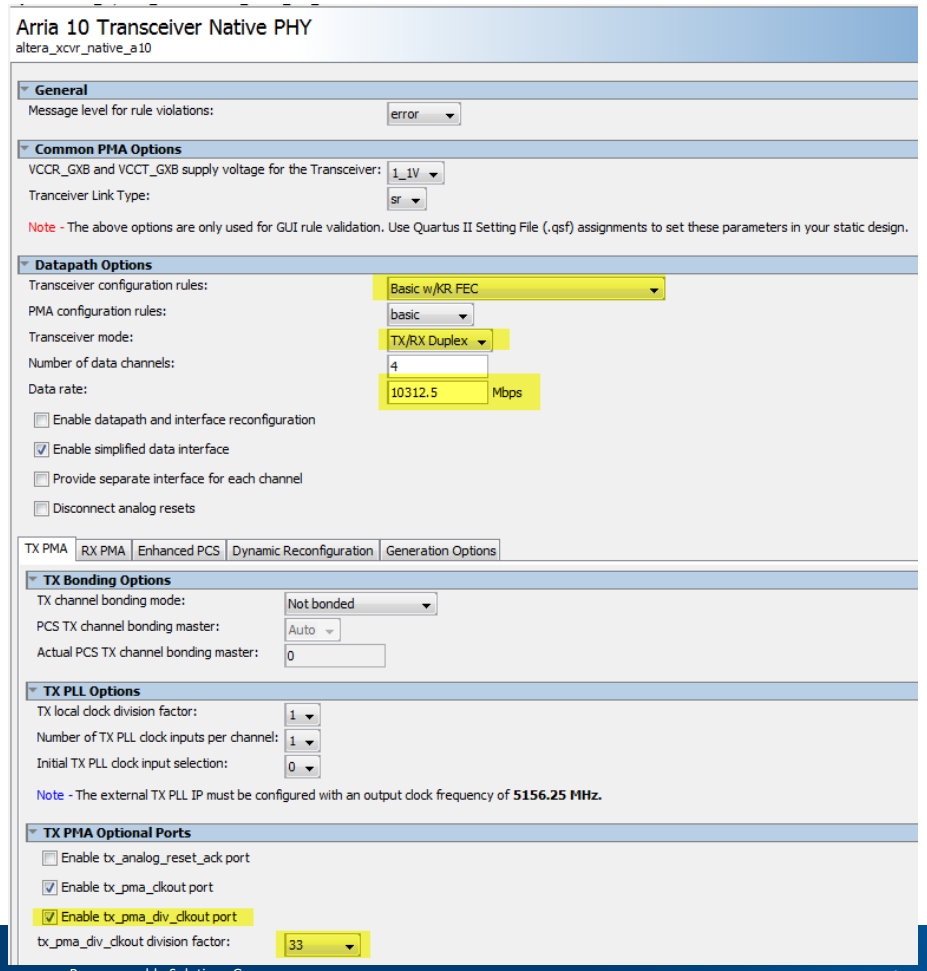
If last control word received is “…5C7C” it means an XOFF has been received from the remote side, this will be forwarded to the local transmitter to stop generating traffic on the local side.

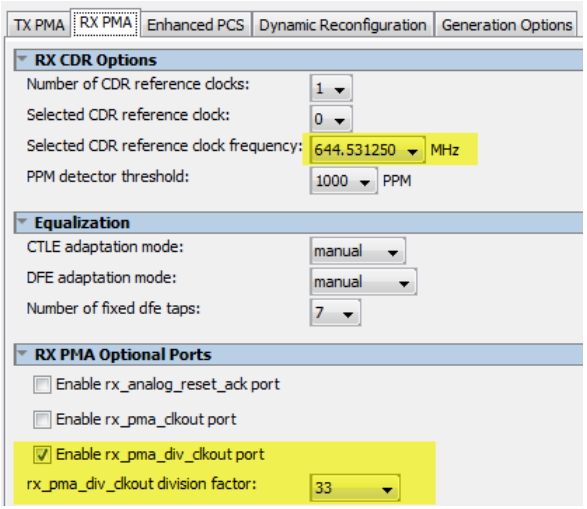
If last control word received is “…3C7C” it means the remote side has stopped sending traffic (as a response to a local XOFF)

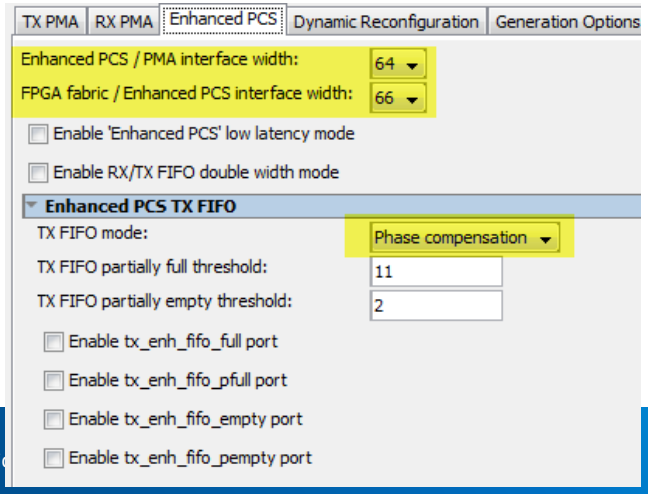
If last control word received is “…9C7C” it means the remote side has stopped sending traffic but at the same time also requesting XOFF. At this point of time both local and remote side are not generating any traffic (Link is still up).

Transceiver Configuration (1/4)

A 4 lanes Duplex Native PHY is selected. Using 10G PCS (Basic w/KR FEC), 66 bit FPGA interface width, 64 bit PCS/PMA width and a datarate of 10312.5 Mbps from a 644.53125 Mhz referenceclock.



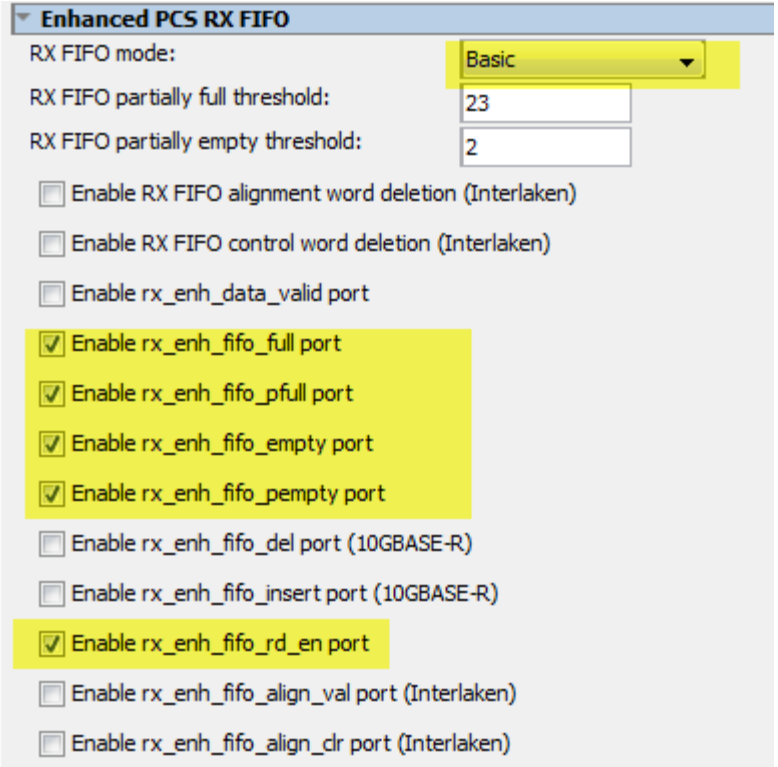




Transceiver Configuration (2/4)

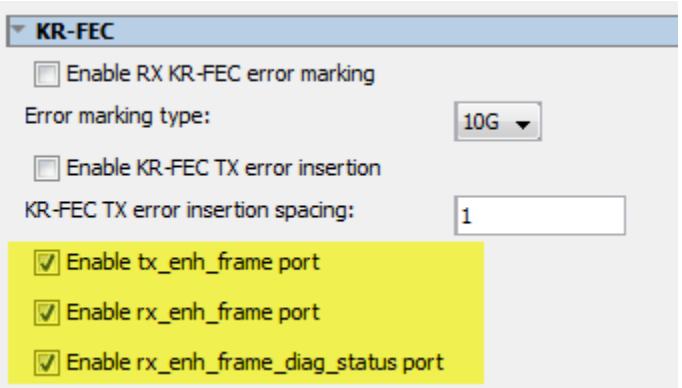
In order to use the Rx FIFO for deskewing the lanes the optional signal “rx_enh_fifo_rd_en” must be enabled (the other fifo status signals are optional).

RX FIFO mode should be set to “Basic”



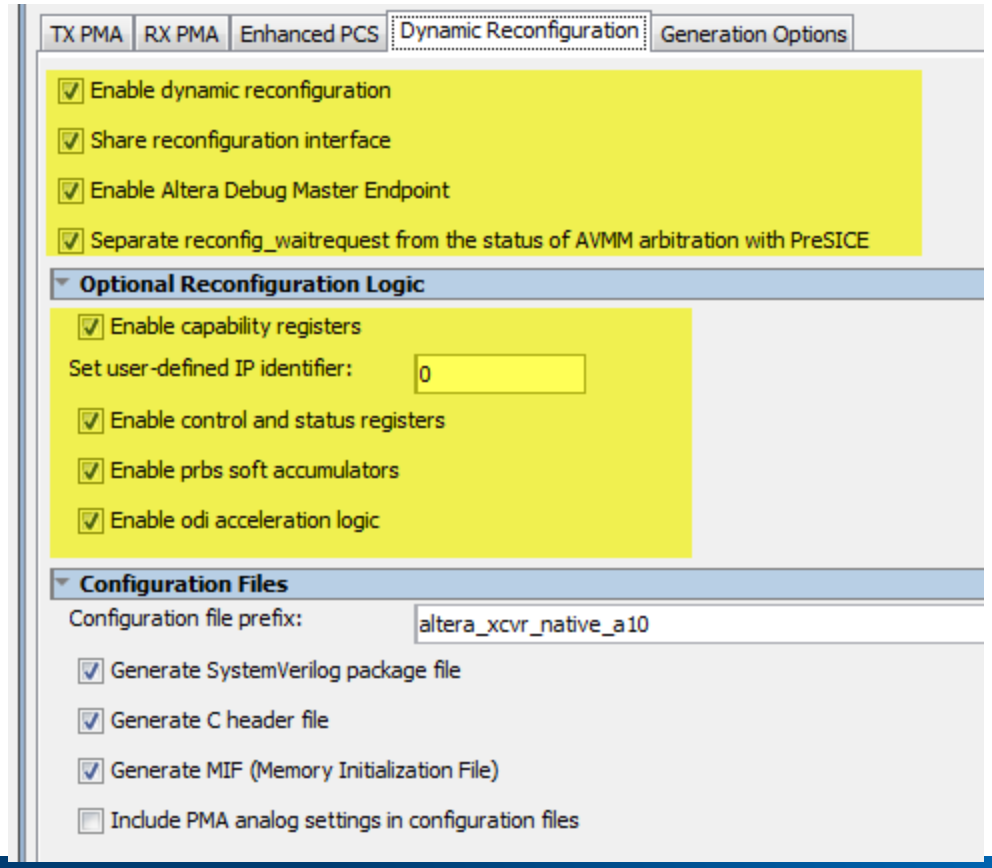
Transceiver Configuration (3/4)

In order to have status information from the FEC Encoder and Decoder the following options are enabled for the KR-FEC section

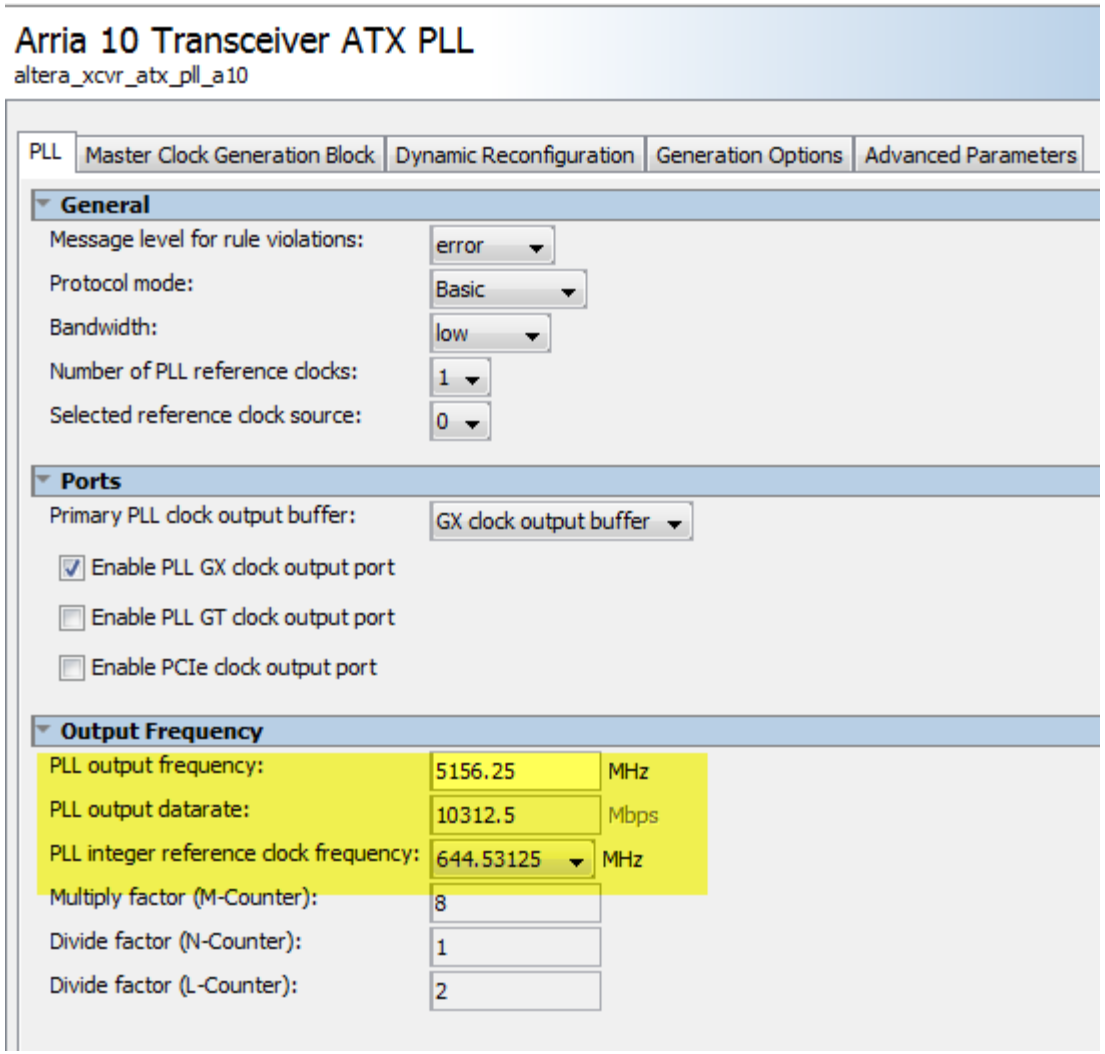


Transceiver Configuration (4/4)

To enable a full set of debug capabilities (including TTK functionality) all the options as shown below are enabled. Note that in a final design these are not all mandatory and can be disabled to save resources.

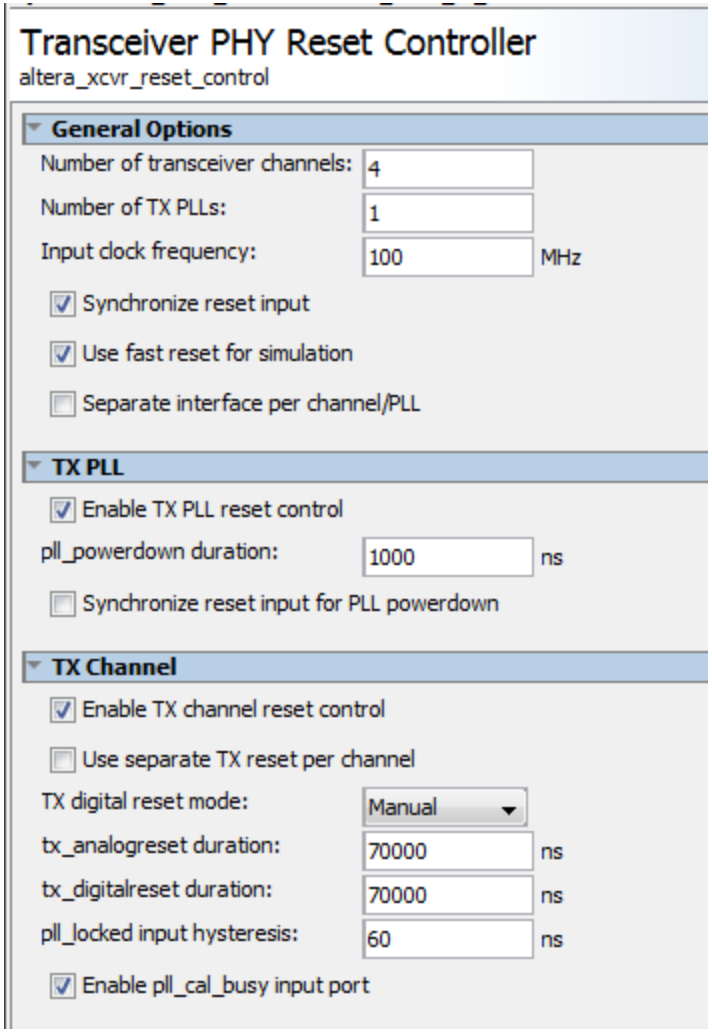


ATX PLL Configuration



Transceiver TX Reset Controller

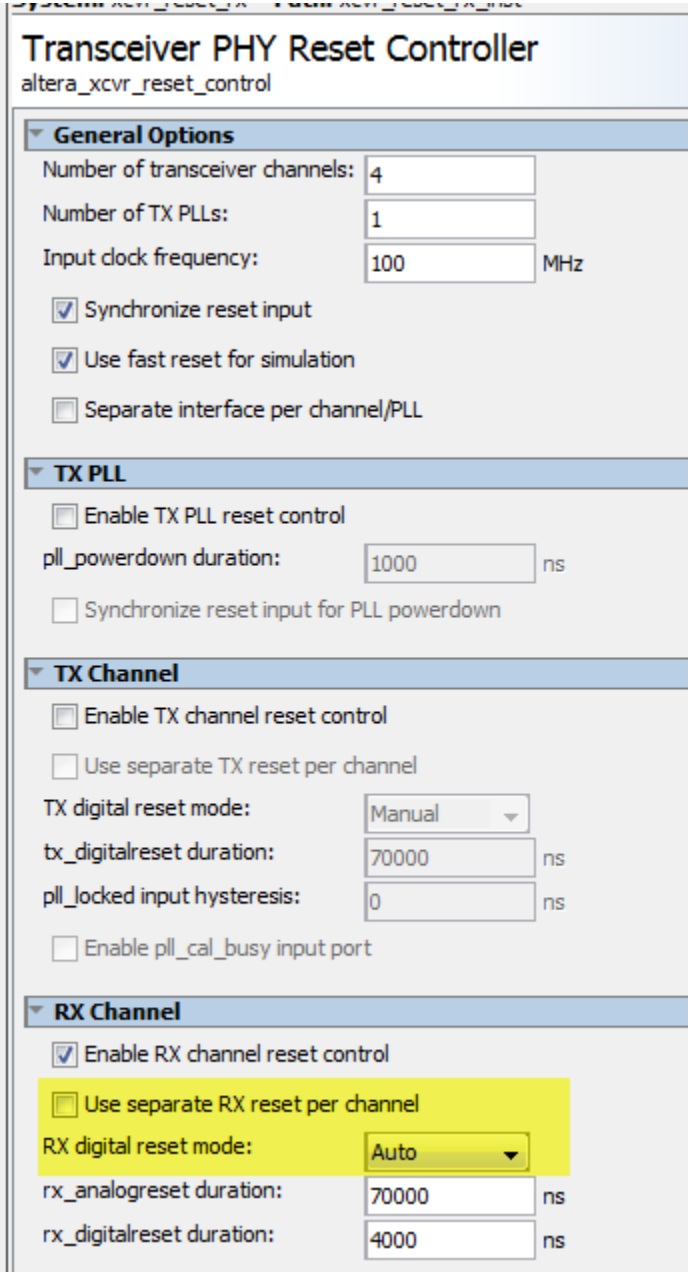
- Uses the recommended timings available in 16.1 release
- Note that the Reset controllers work in combination with the TRS (Transceiver Reset Sequencer) which is automatically inserted by Quartus in the design (transparent to the user).



Transceiver Rx Reset Controller

Uses recommended timings from 16.1 release

Make sure the “Use separate RX reset per channel” is not enabled.



Reconfig Management Interface.

The Native PHY as well as the ATX PLL have an Avalon MM Interface which allow full control of the register space of the transceiver and the ATX PLL.

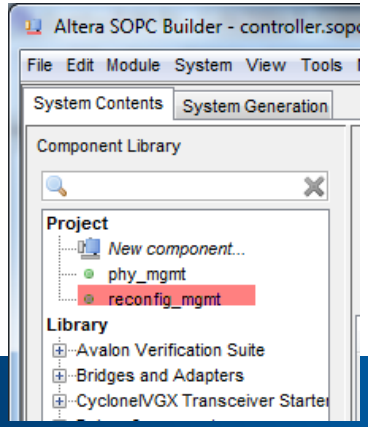
The AVMM used for the Native PHY is using shared addressing for all channels and provides one interface to the .QSYS system.

The AVMM used for the ATX PLL provides another separate interface to the .QSYS system.

This design is using a QSYS system which transparently maps the AMM interfaces to the Native PHY and the ATX PLL.

The component created for this in QSYS is called “reconfig_mgmt” and the reconfig_mgmt_hw.tcl file included in the project describes it’s implementation.

You can verify it’s content by editing the reconfig_mgmt component.



ADME Interface

In addition to the AVMM interface the Native PHY also enables the ADME interface (Altera Debug Master Endpoint) which can also be accessed through the Qsys system due to the addition of the jtag_debug_module.

Access to the ADME interface can be done through System Console totally in parallel while the Nios controller is also being used.

The ADME interface can be used for advanced debugging and reporting (see further) .

The major advantage of having the ADME interface is that it allows to run directly the Transceiver Toolkit (see further).

Timing Closure

The design is fully constraint and has no timing violations

Compiled for 10AX115S2F45I1SG

Slow 900mV 100C Model Setup Summary

<<Filter>>

	Clock	Slack
1	REFCLKGXBL_1E_Tp	0.245
2	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|...native_a10_0|g_xcvr_native_insts[0]|rx_pma_div_clk	1.808
3	LVDS_CLK3p	2.795
4	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[0]|rx_coreclkin	3.036
5	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[2]|rx_coreclkin	3.069
6	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x..._native_a10_0|g_xcvr_native_insts[0]|tx_pma_div_clk	3.110
7	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[3]|tx_coreclkin	3.186
8	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[2]|tx_coreclkin	3.375
9	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[3]|rx_coreclkin	3.403
10	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[1]|rx_coreclkin	3.600
11	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[1]|tx_coreclkin	3.663
12	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...vr_native_a10_0|g_xcvr_native_insts[0]|tx_coreclkin	3.673
13	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...st|xcvr_native_a10_0|g_xcvr_native_insts[0]|avmmclk	4.187
14	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...st|xcvr_native_a10_0|g_xcvr_native_insts[1]|avmmclk	4.323
15	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|...cvr_native_a10_0|g_xcvr_native_insts[0]|rx_pma_clk	4.457
16	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|...cvr_native_a10_0|g_xcvr_native_insts[3]|rx_pma_clk	4.554
17	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|...cvr_native_a10_0|g_xcvr_native_insts[2]|rx_pma_clk	4.566
18	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|...cvr_native_a10_0|g_xcvr_native_insts[1]|rx_pma_clk	4.701
19	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...st|xcvr_native_a10_0|g_xcvr_native_insts[2]|avmmclk	4.995
20	inst2|superliteii_txrx_module_inst|xcvr_txrx_inst|x...st|xcvr_native_a10_0|g_xcvr_native_insts[3]|avmmclk	5.449
21	~ALTERA_CLKUSR~	6.114
22	altera_reserved_tck	7.237

Slow 900mV 100C Model Hold Summary

<<Filter>>

	Clock	Slack
1	LVDS_CLK3p	0.042
2	inst2|superliteii_txrx_module_inst|xc...g_xcvr_native_insts[0]|tx_pma_div_clk	0.042
3	inst2|superliteii_txrx_module_inst|xc...g_xcvr_native_insts[0]|rx_pma_div_clk	0.047
4	altera_reserved_tck	0.048
5	REFCLKGXBL_1E_Tp	0.049
6	~ALTERA_CLKUSR~	0.049
7	inst2|superliteii_txrx_module_inst|xc...0_0|g_xcvr_native_insts[2]|rx_pma_clk	0.051
8	inst2|superliteii_txrx_module_inst|xc...0_0|g_xcvr_native_insts[3]|rx_pma_clk	0.061
9	inst2|superliteii_txrx_module_inst|xc...0_0|g_xcvr_native_insts[1]|rx_pma_clk	0.063
10	inst2|superliteii_txrx_module_inst|xc...0_0|g_xcvr_native_insts[0]|rx_pma_clk	0.082
11	inst2|superliteii_txrx_module_inst|xc..._a10_0|g_xcvr_native_insts[3]|avmmclk	0.408
12	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[2]|tx_coreclkin	0.602
13	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[1]|tx_coreclkin	0.718
14	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[3]|tx_coreclkin	0.737
15	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[0]|tx_coreclkin	0.767
16	inst2|superliteii_txrx_module_inst|xc..._a10_0|g_xcvr_native_insts[2]|avmmclk	1.027
17	inst2|superliteii_txrx_module_inst|xc..._a10_0|g_xcvr_native_insts[1]|avmmclk	1.064
18	inst2|superliteii_txrx_module_inst|xc..._a10_0|g_xcvr_native_insts[0]|avmmclk	1.086
19	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[1]|rx_coreclkin	1.360
20	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[3]|rx_coreclkin	1.391
21	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[0]|rx_coreclkin	1.722
22	inst2|superliteii_txrx_module_inst|xcv..._0|g_xcvr_native_insts[2]|rx_coreclkin	1.773

Nios II Control & Monitoring

The Nios II controller is used to control and monitor the design using the Nios II terminal.

- Display status signals
  - **Freq_Locked**
  - **Pll_locked**
  - **Link Up**
  - **XOFF Received**
  - **DataLocked (combination of the 4 PrbsLocked and Countlocked)**
  - **WordAligned, LaneAligned**
  - **Effective Datarate**
  - **Efficiency**
- Control Serial Loopback on all lanes.
- Control XOFF.
- Generate single biterrors in the generated data (note that inserting an error will result in 11 errors being seen : one per lane.
- Displays the number of biterrors and the BER.

Nios II Control & Monitoring

- Allows to start a continuous BER measurement with a controllable interval (default time interval is 2 seconds).
- Displays the time the test has been running.
- Displays the measured reference clock frequency (Measured with 100 Mhz reference clock during 1 second).
- Control the voltage output level, all 5 pre-emphasis levels and equalization levels of the lanes, on a per lane basis
- Measures the minimum and maximum latency/

Access is through Nios SDK shell (see further)

Nios 2 Output in Nios II SDK Shell

PMA settings used for the result on the right:

Channel	0	1	2	3
VOD	31	31	31	31
Pre Tap 2 Level	0	0	0	0
Pre Tap 1 Level	0	0	0	0
Post Tap 1 Level	0	0	0	0
Post Tap 2 Level	0	0	0	0
CTLE DC Gain	0	0	0	0
CTLE AC Gain	8	8	8	8
VGA Gain	4	4	4	4
Dyn	3	3	3	3
Fix	7	3	3	4
Ipd	2	3	3	3
DFE Enabled	0	0	0	0
DFE Tap 1	0	0	0	0
DFE Tap 2	0	0	0	0
DFE Tap 3	0	0	0	0
DFE Tap 4	0	0	0	0
DFE Tap 5	0	0	0	0
DFE Tap 6	0	0	0	0
DFE Tap 7	0	0	0	0
DFE Tap 8	0	0	0	0
DFE Tap 9	0	0	0	0
DFE Tap 10	0	0	0	0
DFE Tap 11	0	0	0	0
Nios Decision	0	0	0	0

```
Arria 10 GX Devkit Superlite II V3 across  4 Lanes
--------------------------------------------------------------------------------
|Number of Lanes         :  4
|Line rate               : 10312 Mbps
|Aggregrate Line Rate    : 41249 Mbps
|Read Length             : 255
|Idle Length             : 7
|Ratio                   : 0.9725
|Net User Data Bandwidth : 38831 Mbps
|Efficiency              : 94.14 %
|Measured Max Latency    :  364 ns, 57 parallel clocks
|Measured Min Latency    :  262 ns, 41 parallel clocks
|EyeQInterval            :    1 ms
|SelectedChannel         :  0 (QSFP+ Lane 0)
--------------------------------------------------------------------------------

Status
======
Lane            | 0| 1| 2| 3|
                |--|--|--|--|
Serial Loop     : 0| 0| 0| 0|
PLLLocked       : 1
FreqLocked      : 1
Error Deskew    : 0
XOFF Sent       : 0
XOFF Received   : 0
Error Decoder   : 0
WordAligned     : 1
LaneAligned     : 1
LinkUp          : 1
DataLocked      : 1
Errorcount      : 0


BER (CL=0.95) Link         : 3.786787e-12

Test Time                  :  0h  0m 19s
Reference Clock Frequency  :  644526 kHz
DataClock Frequency        :  156248 kHZ
DataClock_out Frequency    :  151686 kHZ
Temperature Arria10        :  32 (degrees Celcius)


Select Action :
===============
1. Set all channels back to default and toggle Reset
2. Force Re-Alignment on Rx Path
3. Select Channel To Control
4. Show/Control Transceiver PMA Settings on Links
5. Insert Biterrors on Link (4 at a time)
6. Reset ErrorCounter
7. Show Status
8. Control Serial loopback
C. Refresh BER every 1 seconds
D. Input new BER Time Interval
E. Show Transceiver PMA Settings on all channels
G. Input new EyeQInterval time
S. Store all channel information in memory of Selected Channel
T. Store and compare with values stored in process S
X. Toggle XOFF to partner to stop/start sending traffic at remote side
Z. Dump channel content of selected channel
O. Perform ODI on selected channel
0. Stop Test

  Enter Choice :
```

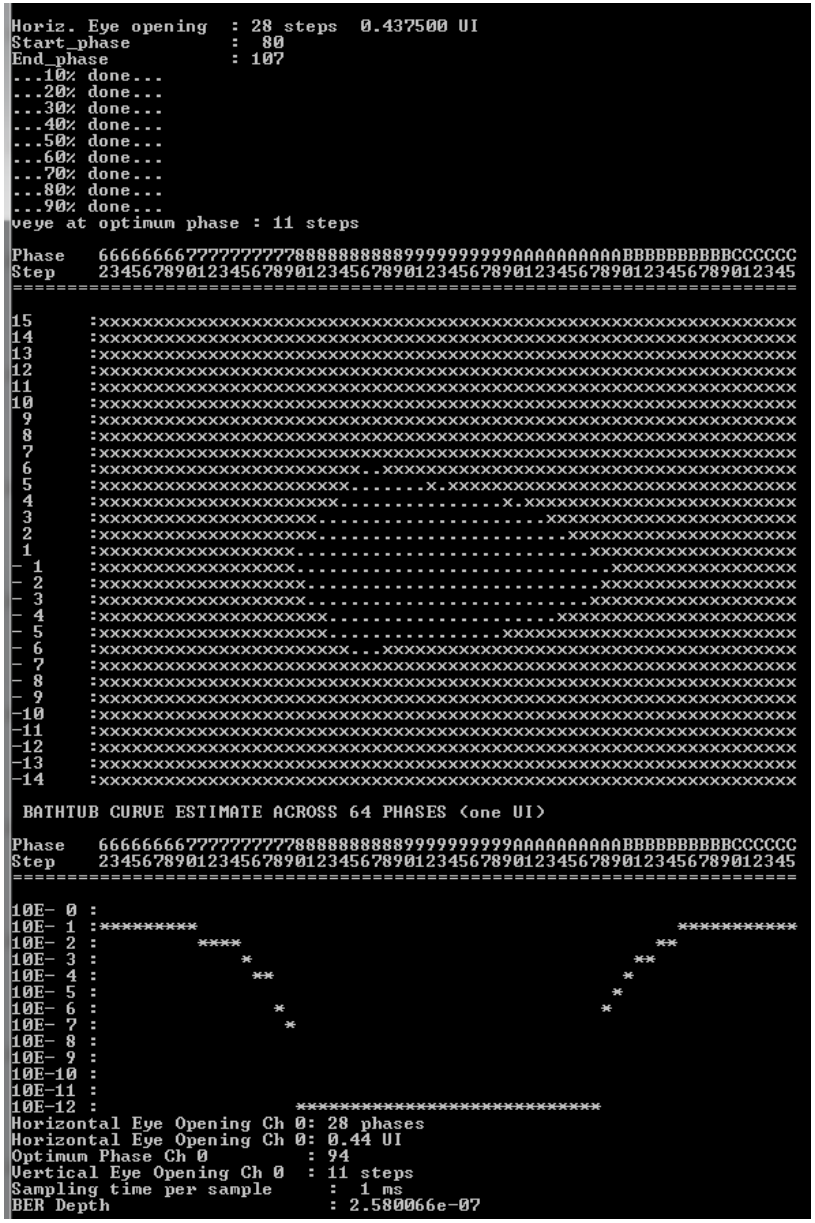
ODI Measurement

The program offers the option to measure both 2D Eye as well as 1D eye on the selected lane.

Time for every measurement point can be dynamically set (Default 1 ms per measurement point)

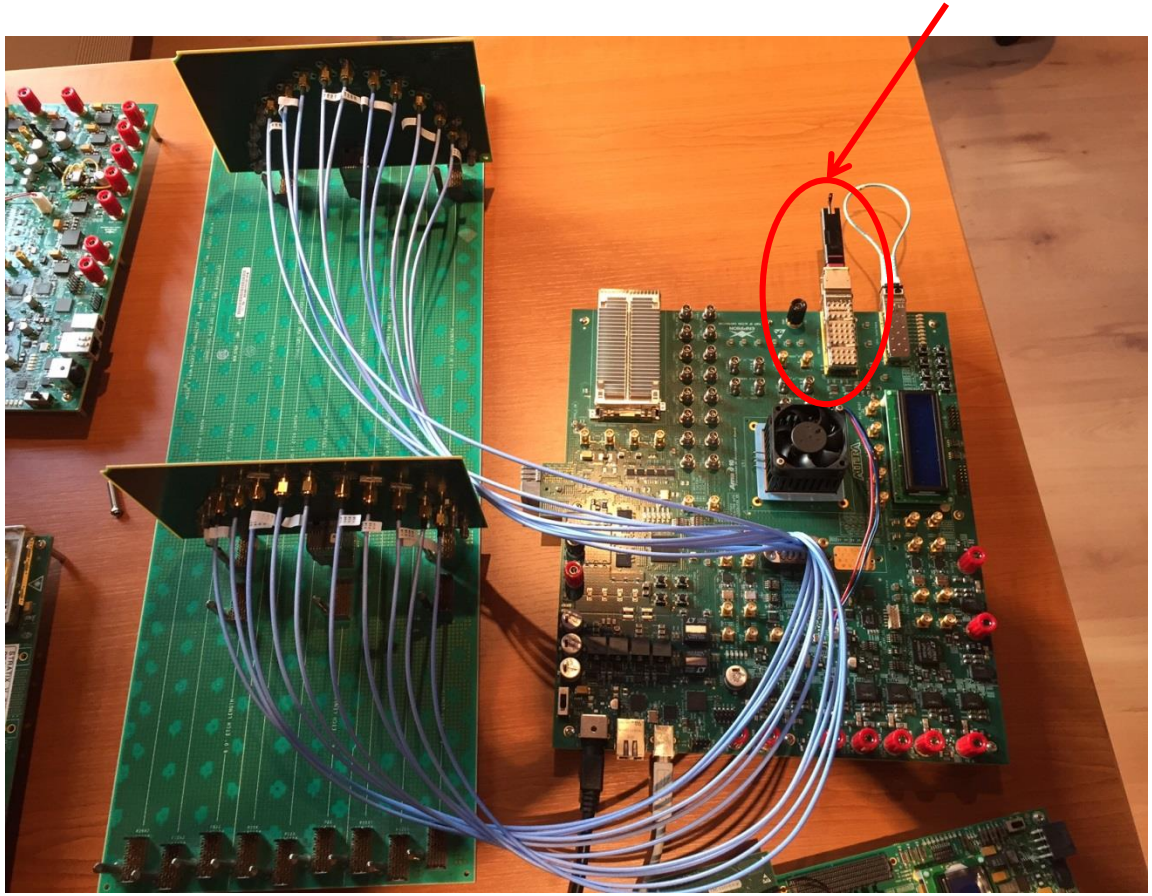
Statistics are being provided including BER depth of the measurement.

Note as this is a very lossy link the measured eye is pretty small. No errors are observed however on this particular lane.



Board Setup

The demo is using 4 transceiver channels routed to the QSFP+ module which is put in optical loopback



Board Setup

Connect USB Cable to CN1 (USB II -Blaster is on the board itself).

Using Onboard oscillator :

- Set CLK_SEL (SW5.1) dipswitch to 'OSC' : No need to reprogram the Si570 (Y3) as it already generates a 644.53125 Mhz clock by default.

Using External Clock :

- Set CLK_SEL (SW5.1) dipswitch to 'SMA' to select the external clock. Connect a 644.53125 Mhz external clock to the External clock inputs

Download 'Devkit_Demo.sof' to the 10AX115S2F45I1SG device using QII programmer or through Signaltap II.

Software Setup

To Start The Nios Controller do the following :

1. Start The Nios II SDK Shell and go to the directory where the .elf file and run.bsh file is located.
2. Type In The Following Command : "./run1.bsh" This will download the Nios II code to the FPGA and open up the terminal.

   Note : in this mode the input provided is not 'echoed' in the terminal, but the value is accepted.

Rebuilding the software

In order to rebuild the software do the following (using QII 16.1 or later):

Start a Nios II 16.1 SDK Shell and go to the ../software/app/Arria10GX_Superlite directory (which is part of the archive) and type: **./create-this-app** .This will compile the system libraries (in the bsp folder), compile the software code (main.c and a number of other .c files in this case) and produce a Nios II executable.

Demonstration on the board

LED's

- LED0 : TxPLL_Locked (Transceiver locked to reference clock
- LED1 : Rx_Freqlocked status of selected channel
- LED2 : WordAligned
- LED3 : LaneAligned
- LED4 : LinkUp
- LED5 : Locked (CountLocked and all PRBSLocked signal asserted)
- LED6: Link OK : Locked + No biterrors
- LED7 : XOFF Received

Signaltap II Debug Example

The design contains a number of signaltap instances, each clocked by it's own clock.

Signaltap II acquisition can be done simultaneously with Nios II control.

Use of ADME (Optional)

To access the ADME, open the System Console.

There is a sample script in the scripts folder which has a couple of useful debug procedures.

It can be started through “source ttk_helper.tcl” and provides a list of procedures available and the syntax to use.

Using TTK_Helper.tcl script

When the system console is open you can source the ttk_helper.tcl script located in the scripts folder. This script contains a number of useful routines for advanced debug and setup.

A particular interesting procedure is one which set generator and checker to PRBS-31 and starts the tests.

To run this prodedure run : “start_check_prbs31”

See the attached file for an overview of available procedures. Including functions to do reverse serial loopback, polarity inversion, calibration, enabling, disabling dfe etc.

functions.txt

Using Transceiver Toolkit (Optional)

To start the Transceiver Toolkit simply start it from the Tools Menu in Quartus (with the project loaded). Using 16.1 or later.

This will automatically detect the design and start up the Transceiver Toolkit which allows you to run further tests in optimizing the PMA settings, do autosweeps etc.

So you can e.g. run an autosweep using TTK to find optimum settings and when you go back to the original design (after pressing '1') all the PMA settings as entered in the TTK will be kept and used. Vice Versa all PMA settings done through the Nios design will also be read out automatically and used by the TTK.

Once finished with the TTK make sure all tests are stopped and after quitting the TTK and going back to the original design, press "1" to reset the design again.

Note : in order to enable EyeQ capability in the TTK you need to add the following lines in your quartus2.ini file (not the one in the project directory) but located here "c:\users\<Your User name>\"

```
[TTK INTERNAL]
ttk_a10_allow_dfe_eyeq = on
ttk_a10_eyeq = on
```

TestBench

First use : Open a Nios II 16.1 Shell and go to the simulation directory. In the shell provide the following command “./create_simscript.bsh”. This will create the necessary simulation scripts to match your own environment.

When using Modelsim SE (or PE) or Questasim (tested with QuestaSim 10.5c)

- Run ‘Sim.do’ do compile and simulate everything (when there are no libraries yet)
- Run ‘Resim.do’ when libraries already exist to re-compile only the design files and simulate the testbench

The simulation files can be found in the ‘simulation’ directory

After lane alignement has been achieved the testbench will also insert 2 errors (each resulting in 4 biterrors) which are reported by the testbench.

Deliverables

Quartus 16.1.2 archive which contains the demo design files, the software source files, ttk_helper.tcl scripts and the testbench files.

The testbenches can be found after restoring the archive: <restored_project>/simulation

The software source files can be found after restoring the archive in the <restored_project>/software/app and <restored_project>/software/bsp folders/

The software executables.

SOF File.

ttk_helper.tcl file in the /scripts folder

This document.

Status & Revision

V16.1.0 : Initial release

- Succesfully tested with QSFP+ optical loopback module installed.

Feedback and/or Questions?

Peter Schepers (peter.schepers@intel.com)

+32 495 57 32 22