XAPP1240 (v3.1) November 4, 2022

Clock and Data Recovery Unit based on Deserialized Oversampled Data

Authors: Paolo Novellini, Antonello Di Fresco, and Giovanni Guasti

Summary

Multi-service networks require transceivers that can operate over a wide range of input data rates. High-speed serial I/O has a native lower limit data rate that prevents easy interfacing to low-speed client signals. The non-integer data recovery unit (NIDRU) presented in this application note extends the lower data rate limit to 0 Mb/s, making dedicated high-speed transceivers the ideal solution for true multirate serial interfaces. The NIDRU is specifically designed for the Xilinx® 7 series, UltraScale™, and Versal® devices. The NIDRU operational settings (data rate, jitter bandwidth, input ppm range, and jitter peaking) are dynamically programmable, avoiding the need for bitstream reload or partial reconfiguration. Operating on a synchronous external reference clock, the NIDRU supports fractional oversampling ratios. Thus, only one clock tree (BUFG or BUFG_GT) is needed, and is independent of the number of channels being set up, even if all channels are operating at different data rates.

The extracted data is delivered to the user application in parallel format. The width of this bus is programmable, thus simplifying the connection to both 8-bit and 10-bit applications.

At any time, without affecting the data traffic, a live horizontal eye scan can be performed to measure the real eye width as seen by the NIDRU. An eye scan controller, managing one or two extra phases, is embedded into the NIDRU wrapper.

You can download the reference design files for this application note from the Xilinx website. For detailed information about the design files, see Reference Design.

Introduction

This application note is divided into four sections:

- NIDRU Block Diagram and Pinout
  Describes the structure of the NIDRU wrapper and its pinout.
- NIDRU Usage Model
  Describes how to configure the NIDRU ports and attributes.
- Simulating the NIDRU
  Describes the test bench used to simulate the DRU.
- FPGA Hardware Test Bench
  Describes the TB_HW_DRU test bench, available in the reference design.

The Fast Ethernet case (125 Mb/s ± 100 ppm) and OC3/STM1 (155.520 Mb/s ± 20 ppm) are used as practical examples throughout the application note.

NIDRU Block Diagram and Pinout

This section describes the structure of the NIDRU wrapper and its pinout. The wrapper structure is shown in Figure 1. Only relevant ports are shown.

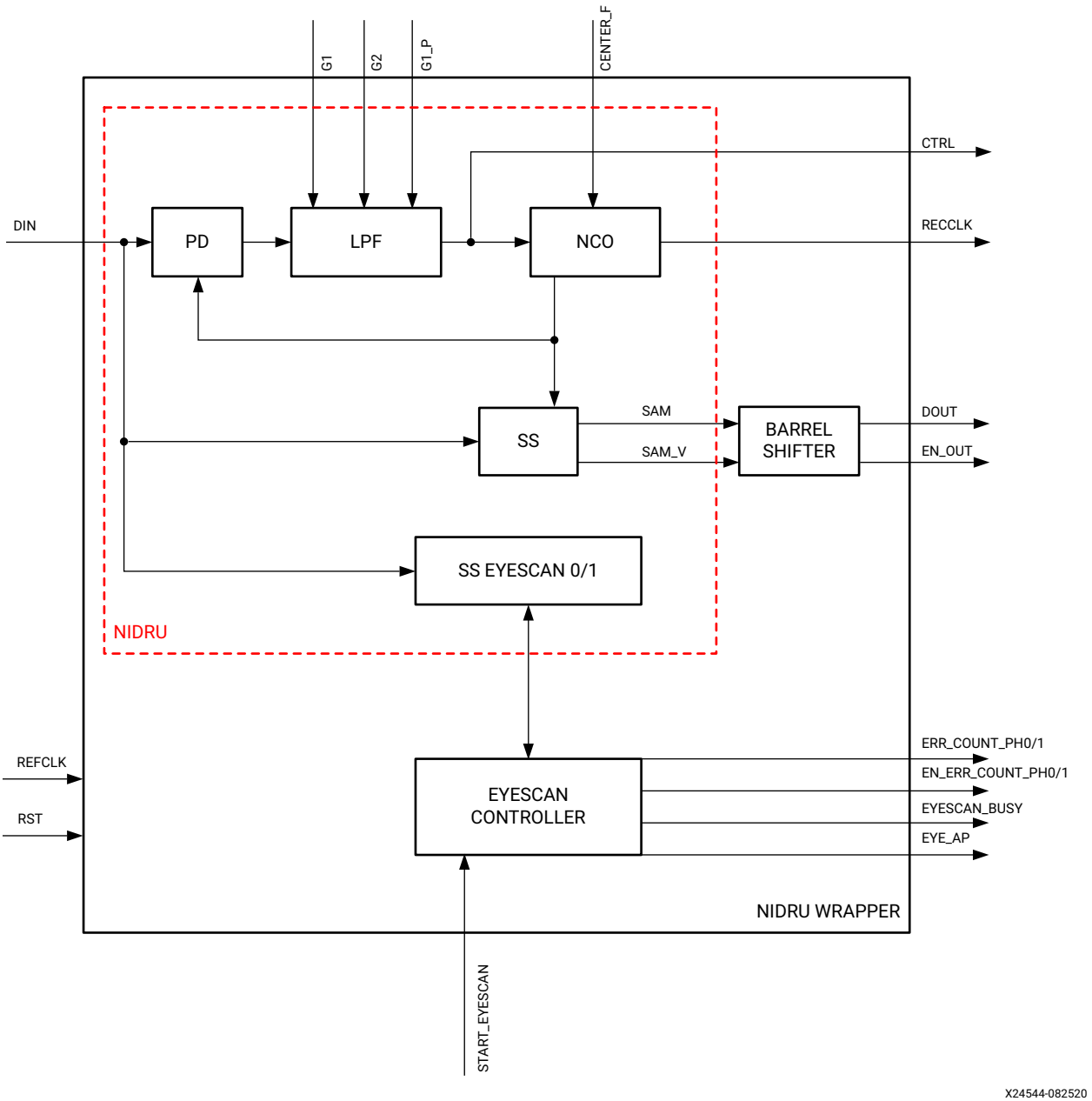


Figure 1: **NIDRU Wrapper Block Diagram**

The DIN port receives raw oversampled data from a SelectIO™ interface or a SerDes set in lock-to-reference mode. The width of the oversampled data is programmable by the

DT_IN_WIDTH attribute. The value of the DT_IN_WIDTH attribute can be set to 4, 20, or 32 bits. The NIDRU bit-ordering convention is the same as the SerDes, where the LSB is the oldest bit(1). The phase detector (PD) looks for transitions in the incoming data, continuously comparing the phase of the incoming data with the phase of the internal numerically-controlled oscillator (NCO). The digital error signal generated by the PD and filtered out by the low-pass filter (LP) corrects the NCO frequency to minimize the phase error, thus realizing the phase-locked loop (PLL) functionality of the NIDRU [Ref 1] [Ref 2]. Based on the NCO output, the sample selector block (SS) selects the samples that are more closely positioned to the middle of the eye. There can be up to 10 valid samples in each REFCLK cycle, which are placed by the SS on the SAM output. SAMV indicates the number of valid samples on SAM at each clock cycle. To simplify the connection between the NIDRU and the user application, a barrel shifter is provided in the wrapper, where the output data width can be programmed using the WDT_OUT attribute. All blocks in the NIDRU wrapper are synchronized to REFCLK.

The NIDRU operates in parallel over the incoming data, generally producing more than one bit output for each clock cycle. The relationship between the operating frequency (REFCLK) and the incoming data rate dictates the maximum number of bits processed per clock cycle, N_{MAX}, according to Equation 1:

$$N_{MAX} = truncate\left[\frac{f_{DIN}}{f_{REFCLK}}\right] + 1 \qquad \textit{Equation 1}$$

Three example user configurations of f_{REFCLK} and oversampling rate are considered here and yield these results for N_{MAX}:

- Fast Ethernet with f_{REFCLK} = 125 MHz, oversampling at 2.5 Gb/s: N_{MAX} = 2.
- STM1 with f_{REFCLK} = 125 MHz, oversampling at 2.5 Gb/s: N_{MAX} = 2.
- Fast Ethernet with f_{REFCLK} = 155.52 MHz, oversampling at 3.1 Gb/s: N_{MAX} = 1.

If $N > 1$, the barrel shifter eases the interfacing of the DRU to a fixed-width FIFO. If $N = 1$ and the user application has 1-bit width only, the barrel shifter is not needed.

Table 1 describes the NIDRU configuration attributes. NIDRU ports are described in Table 2.

Table 1: **NIDRU Configuration Attributes**

Attribute Name	Type/Range	Description	Comment
Configuration Section			
WDT_OUT	Integer from 2 to 64	Output data width	Output width for the bus DOUT.
DT_IN_WIDTH	Integer 4, 20, 32, 64, or 128	Input data width	Output width for the bus DT_IN.
EN_CENTER_F_ATTR	Standard logic	Enables use of Center_f_attr	When set to 1, CENTER_F_ATTR is used as CENTER_F. When set to 0, the CENTER_F port is used.

1. Starting with Virtex®-4 devices, all SerDes follow the same ordering rule as the NIDRU described here. SerDes in Virtex-II Pro devices use a different bit ordering.

Table 1: **NIDRU Configuration Attributes *(Cont'd)***

Attribute Name	Type/Range	Description	Comment
CENTER_F_ATTR	Standard logic vector 36 down to 0	Attribute configuration for CENTER_F	CENTER_F_ATTR can be used instead of CENTER_F depending on the value of EN_ CENTER_F_ATTR.
EN_G1_ATTR	Standard logic	Enables use of G1_ATTR	When set to 1, CENTER_F_ATTR is used as CENTER_F. When set to 0, the CENTER_F port is used.
G1_ATTR	Standard logic vector 4 down to 0	Attribute configuration for G1	G1_ATTR can be used instead of G1 depending on the value of EN_ G1_ATTR.
EN_G2_ATTR	Standard logic	Enables use of G2_ATTR	When set to 1, CENTER_F_ATTR is used as CENTER_F. When set to 0, the CENTER_F port is used.
G2_ATTR	Standard logic vector 4 down to 0	Attribute configuration for G2	G2_ATTR can be used instead of G2 depending on the value of EN_G2_ATTR.
EN_G1_P_ATTR	Standard logic	Enables use of G1_P_ATTR	When set to 1, CENTER_F_ATTR is used as CENTER_F. When set to 0, the CENTER_F port is used.
G1_P_ATTR	Standard logic vector 4 down to 0	Attribute configuration for G1_P	G1_P_ATTR can be used instead of G1_P depending on the value of EN_G1_P_ATTR.
EN_SHIFT_S_PH_ATTR	Standard logic	Enables use of SHIFT_S_PH_ATTR	When set to 1, CENTER_F_ATTR is used as CENTER_F. When set to 0, the CENTER_F port is used.
SHIFT_S_PH_ATTR	Standard logic vector 7 down to 0	Attribute configuration for SHIFT_S_PH	SHIFT_S_PH_ATTR can be used instead of SHIFT_S_PH depending on the value of EN_SHIFT_S_PH_ATTR.
EN_EN_INTEG_ATTR	Standard logic	Enable port EN_INTEG	When set to 1 enables EN_INTEG port. When set to 0, the EN_INTEG port is connected to EN_INTEG_ATTR.
EN_INTEG_ATTR	Standard logic	Attribute configuration for EN_INTEG	See EN_EN_INTEG_ATTR.
EN_EN	Standard logic	Enables the EN port	The EN port can be disabled by setting EN_EN=1, reducing the complexity of the circuit.
ENABLE_LTR_PORT	Standard logic	Enables lock to reference mode	When set to 1, the port LTR can be used to disable the tracking mechanism (LTR = 1).
Eye Scan Section			
PH_NUM	Integer from 0 to 2	Number of extra sampling phases	0: No eye scan logic is instantiated. 1: Eye scan logic with one extra phase. 2: Eye scan logic with two extra phases.
Logic Optimization Section			
S_MAX	Integer from 1 to 64	Expected maximum number of extracted samples per clock cycle	See NIDRU Usage Model, page 7 for configuration instructions for this port. Set S_MAX ≥ N_{MAX}. Setting S_MAX to 2 if DT_IN_WIDTH = 4 or to 10 if DT_IN_WIDTH = 20 or to 16 if DT_IN_WIDTH = 32 works for all cases.

Table 1: **NIDRU Configuration Attributes *(Cont'd)***

Attribute Name	Type/Range	Description	Comment
S_MAX_EYE	Integer from 1 to 64	Maximum number of samples extracted by the Eyescan controller	See NIDRU Usage Model, page 7 for configuration instructions for this port.
MASK_CG	Standard logic vector 15 down to 0	Mathematical precision of the generated coefficients	See NIDRU Usage Model, page 7 for configuration instructions for this port. Setting MASK_CG to all ones forces NIDRU to use maximum precision.
MASK_PD	Standard logic vector 15 down to 0	Mathematical precision of the PD calculations	See NIDRU Usage Model, page 7 for configuration instructions for this port. Setting MASK_PD to all ones forces NIDRU to use maximum precision.
MASK_VCO	Standard logic vector 36 down to 0	Mathematical precision of the NCO output	See NIDRU Usage Model, page 7 for configuration instructions for this port. Setting MASK_VCO to all ones forces NIDRU to use maximum precision.

Table 2 describes the NIDRU ports. NIDRU configuration attributes are described in Table 1.

Table 2: **NIDRU Ports**

Pin Name	Type	Description	Comment
Data Ports			
DT_IN	Input 4, 20, 32, 64, or 128 bits wide	Input data from SerDes or SelectIO interface	Bit 0 is the oldest.
EN	Input	Enable	Enables all processes of the NIDRU.
CLK	Input	Clock	Clock for all NIDRU processes.
RST_FREQ	Input	Integral path reset	Resets the integral path in the NIDRU.
RECCLK	Output DT_IN_WIDTH bits	Recovered clock	This is the recovered clock to be serialized by a TX SerDes or a SelectIO interface. In terms of serialization order, bit 0 has to be serialized first.
EN_OUT	Output	Output data valid	When data on DOUT is valid, the NIDRU sets EN_OUT to 1.
DOUT	Output WDT_OUT bits	Output data	Output data for the user application. The width of DOUT is programmable through the attribute WDT_OUT.
Config Ports			
CENTER_F	Input 37 bits	Center frequency at which the NIDRU operates	See NIDRU Usage Model, page 7 for configuration instructions for this port.
G1	Input 5 bits	Direct gain	See NIDRU Usage Model, page 7 for configuration instructions for this port.
G1_P	Input 5 bits	Integral pre-gain	See NIDRU Usage Model, page 7 for configuration instructions for this port.
G2	Input 5 bits	Integral post-gain	See NIDRU Usage Model, page 7 for configuration instructions for this port.

Table 2: **NIDRU Ports *(Cont'd)***

Pin Name	Type	Description	Comment
LTR	Input	Lock to reference mode	See attribute EN_LTR_PORT.
Eye Scan Ports			
AUTOM	Input	Auto/manual mode	Setting to 1 enables the embedded eye scan controller. Setting to 0 allows performing eye scan manually.
START_EYESCAN	Input	Start eye scan	Pulse for at least 1 clock cycle to request an eye scan. Any eye scan request while EYESCAN_BUSY = 1 is discarded.
EYESCAN_BUSY	Output	Eye scan is operating	When set to 1, eye scan is being acquired.
TRIGGER_MODE	Input	Eye acquisition mode	See One-Dimensional Eye Scan.
EYE_AP	Output	The eye aperture can be read here.	This value is updated each time the signal EYSCAN_BUSY goes down.
RST_PH_0	Input	Phase 0 reset	See One-Dimensional Eye Scan.
RST_PH_1	Input	Phase 1 reset	
RST_PH_SAMP	Input	Sampling phase reset	
ERR_PH_0	Output 7 bits	Error number from phase 0	
ERR_PH_1	Output 7 bits	Error number from phase 1	
PH_0	Input 8 bits	Phase 0 position	
PH_1	Input 8 bits	Phase 1 position	
PH_0_SCAN	Output 8 bits	Current phase 0 being scanned	
PH_1_SCAN	Output 8 bits	Current phase 1 being scanned	
WAITING_TIME	Input 48 bits	Waiting time for each sample	
ERR_COUNT_PH_0	Output 52 bits	Errors accumulated in PH_0	
EN_ERR_COUNT_PH_0	Output	Valid signal for ERR_COUNT_PH_0	
ERR_COUNT_PH_1	Output 52 bits	Errors accumulated in PH_1	
EN_ERR_COUNT_PH_1	Output	Valid signal for ERR_COUNT_PH_1	
Debug Ports			
PH_OUT	Output	NCO phase output	Debug outputs.
INTEG(31:0)	Output	Integral-branch output	
DIRECT(31:0)	Output	Direct-branch output	
CTRL(31:0)	Output	NCO control signal	
AL_PPM	Output	PPM alarm	When set to 1, the input frequency has exceeded the range for which NIDRU has been configured. This signal is not latched.

Table 2: **NIDRU Ports *(Cont'd)***

Pin Name	Type	Description	Comment
RST	Input	Reset	Resets NIDRU, except for the integral path.
PH_EST_DIS	Input	Phase error estimation method	Debug input. Set to 0.
EN_INTEG	Input	Enable for the integral path	Debug input. Set to 1.
VER	Output 8 bits	Version	NIDRU version. The version delivered with this application note is 9.
SAMV	Output 7 bits	Number of samples out	At each clock cycle, the NIDRU reports how many bits have been extracted. SAMV is connected to the barrel shifter in the wrapper.
SAM	Output 10 or 16 bits	Samples out	At each clock cycle, NIDRU reports the SAMV bits which have been extracted. They are placed in the lowest portion of SAM. SAM is connected in the wrapper to the barrel shifter.
CF_ADD	Input, debug	Set to 0.	Reserved.
SHIFT_S_PH	Input 8 bits	Sampling phase shift	Reserved. Adjusts the position of the sampling phase inside the eye. Resolution is 256 steps. Default is 0.

NIDRU Usage Model

This section describes the usage model and how to configure the NIDRU ports and attributes based on the user application requirements. The usage model describes how to algorithmically size the hardware parameters G1, G2, G1_P, and CENTER_F.

The general clocking structure around the NIDRU is shown in Figure 2. Two clock domains are highlighted, remote and local. The frequency is indicated in brackets for each clock domain.

IMPORTANT: *The NIDRU clock is always locked to the PHY clock, and is thus part of the same clocking domain. In most cases, the ratio between the two clocks is 1.*

Figure 2 is a high-level diagram where the divide/multiply function is typically performed internally in the SerDes block, by a PLL, or by a general interconnect divider in conjunction with the EN signal of the NIDRU.

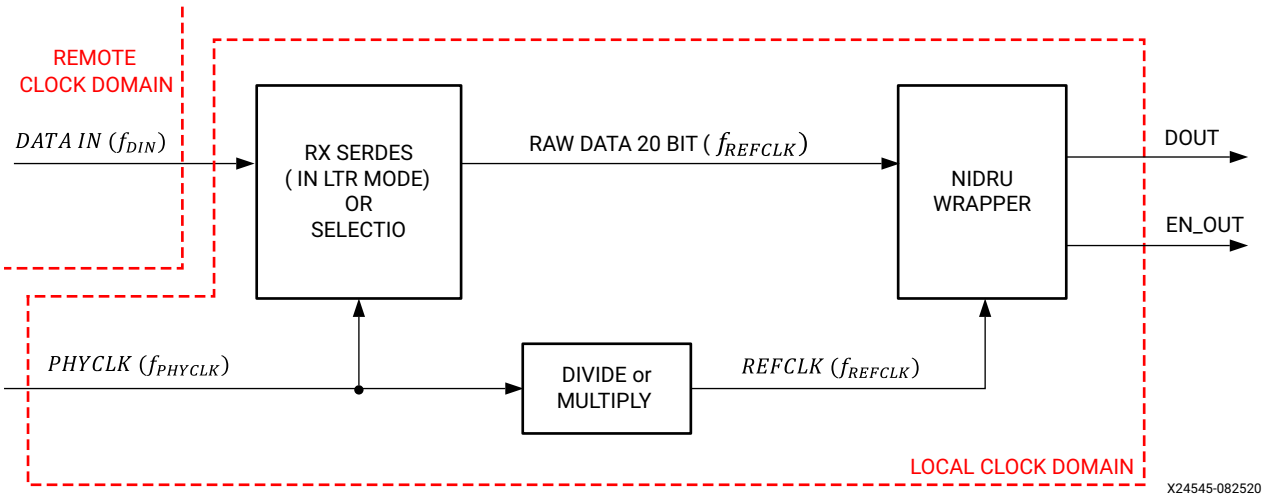


Figure 2: **NIDRU Clocking Structure**

EN_OUT is synchronized to the local clock. However, the rate at which EN_OUT is set to 1 by the NIDRU is locked to the remote clock domain, filtered by the phase transfer function performed by the NIDRU.

The key feature of the NIDRU is that the ratio between the remote clock domain and the local clock domain can be fractional, and this ratio is specified using CENTER_F.

NIDRU Configuration

This section describes how to translate the incoming data rate and reference clock frequency into a valid NIDRU configuration.

The user configuration defines specifications for:

- Incoming data rate with associated tolerance (f_{DIN} ± ppm)
- Available reference clock frequency with associated tolerance (f_{REFCLK} ± ppm)

While f_{DIN} is given, f_{REFCLK} can be selected inside a valid range. The range upper limit comes from the necessity to close timing in the target device, and is thus device and speed grade dependent. The lower limit to f_{REFCLK} might be imposed by the PHY. For example, a SerDes typically specifies a minimum reference clock frequency. A SelectIO interface does not typically impose a lower limit to f_{REFCLK}.

The maximum f_{DIN} is typically limited by the oversampling rate O_R defined in Equation 2:

$$O_R = \frac{DT\_IN\_WIDTH \cdot f_{REFCLK}}{f_{DIN}} \qquad \textit{Equation 2}$$

Although the O_R has to be at least > 2, Xilinx recommends to keep $O_R \geq 3$ to have enough high frequency jitter tolerance.

The theoretical high frequency jitter tolerance is related to O_R as defined in Equation 3:

$$J_{(TOL-HF)} = 1 - \frac{1}{O_R} \quad \text{Equation 3}$$

All DRUs for which Equation 3 is valid have an optimal $J_{(TOL-HF)}$.

Use Equation 4 and Equation 5 to calculate the NIDRU parameters *CENTER_F*, G_1, and G_2. Equation 4 calculates *CENTER_F:*

$$CENTER\_F = \frac{f_{DIN}}{f_{REFCLK}} \times 2^{32} \quad \text{Equation 4}$$

Equation 5 calculates G_1 and G_2:

$$G_1 = G_2 \leq 32 - roundup\left[\log_2 \frac{2^{33} \cdot (ppm_{DIN} + ppm_{REFCLK}) \cdot f_{DIN} \cdot 10^{-6}}{f_{REFCLK}}\right] \quad \text{Equation 5}$$

The spreadsheet `nidru_transfer_function_v_1_0.xls` in the reference design folder `/excel_plots` implements the equations listed above.

Using an equal value for G_1 and G_2 guarantees that the NIDRU operates in the lock-in region over the full tolerance range (ppm) of both the incoming data and the reference clock. Further reducing G_2 increases the NIDRU bandwidth. Increasing G_2 is not recommended because the NIDRU would operate in the pull-in region where the automatic lock is not always guaranteed.

G_{1_P} can be evaluated using the spreadsheet `nidru_transfer_function_v_1_0.xls` in the reference design folder `/excel_plots`. The G_{1_P} value should be increased until the ringing effect on the output phase becomes negligible. When $G_1 = G_2$, setting $G_{1\_P} = 16$ guarantees a negligible ringing effect. Thus, $G_{1\_P} = 16$ is good for most cases.

Examples

Three configurations of f_{DIN} and f_{REFCLK} are considered here and yield these results for CENTER_F, G_1, and G_2:

- Fast Ethernet (f_{DIN} = 125 Mb/s ± 100 ppm with f_{REFCLK} = 125 MHz ± 100ppm): CENTER_F = `b00001000000000000000000000000000` and $G_1 = G_2 \leq 11$.
- Fast Ethernet (f_{DIN} = 125 Mb/s ± 100 ppm with f_{REFCLK} = 155.52 MHz ± 20 ppm): CENTER_F = `b00000110011011100001011100101101001` and $G_1 = G_2 \leq 11$.
- OC3 (f_{DIN} = 155.52 Mb/s ± 20 ppm with f_{REFCLK} = 125 MHz ± 100 ppm): CENTER_F = `b00001001111101000000101000101000011110` and $G_1 = G_2 \leq 11$.

Logic Optimization

The NIDRU performs many calculations during runtime. The precision of these calculations is controlled by the MASK_CG attribute, which permits the ability to trade off between precision and resource usage. For example:

- MASK_CG = `1111111111111111` sets calculations to internal 16-bit precision (highest precision, highest resource usage)
- MASK_CG = `1111111111111110` sets calculations to internal 15-bit precision (lower precision, lower resource usage)

TIP: *Lowering the internal precision to 10-bit has a negligible impact on the performance.*

Reducing the internal calculation precision negatively impacts the jitter tolerance. The amount has to be evaluated either in hardware or in simulation.

S_MAX is the expected maximum number of extracted samples per clock cycle. N_{MAX} is the maximum number of bits processed per clock cycle.

IMPORTANT: *It is mandatory to set S_MAX ≥ N_MAX. N_MAX is calculated using the following equation: N_MAX = truncate (fDIN/fREFCLK+1)*

S_MAX_EYE is an internal parameter and must be set according to the following equation:

$$S\_MAX\_EYE = S\_MAX + 1 \qquad \textit{Equation 6}$$

One-Dimensional Eye Scan

The NIDRU allows plotting the eye diagram using live data while not affecting the data traffic. This is a useful debugging feature because it allows measuring the sampling margins. Also, during normal operation, this feature is useful for detecting links that are degrading over time, before the degradation can result in a visible BER.

The resolution of the eye scan is 256 taps for a single UI, independent of the oversampling rate.

The PH_NUM attribute is used to activate and configure the eye scan logic:

- PH_NUM = 0 = no eye scan
- PH_NUM = 1 = the eye scan operates with one exploring phase, ranging from –128 to 127, equivalent to –0.5 UI to 0.5 UI
- PH_NUM = 2 = the eye scan operates with two exploring phases using the automatic eye scan controller. Ph_0 sweeps the right portion of the eye from 0 up to 127 (from 0 to 0.5 UI). PH_1 sweeps the left side of the eye from –1 to –128.

In automatic mode (AUTOM = 1), all steps to perform an eye scan are managed by the embedded eye scan controller and each sweep can be triggered by pulsing the START_EYESCAN port for at least one clock cycle. The output EYESCAN_BUSY is set to 1 until all 256 points are swept. The port WAITING_TIME is used to specify how many clock cycles should be devoted to each single sweeping point.

Figure 3 shows the eye scan controller timing diagram when PH_NUM = 1.

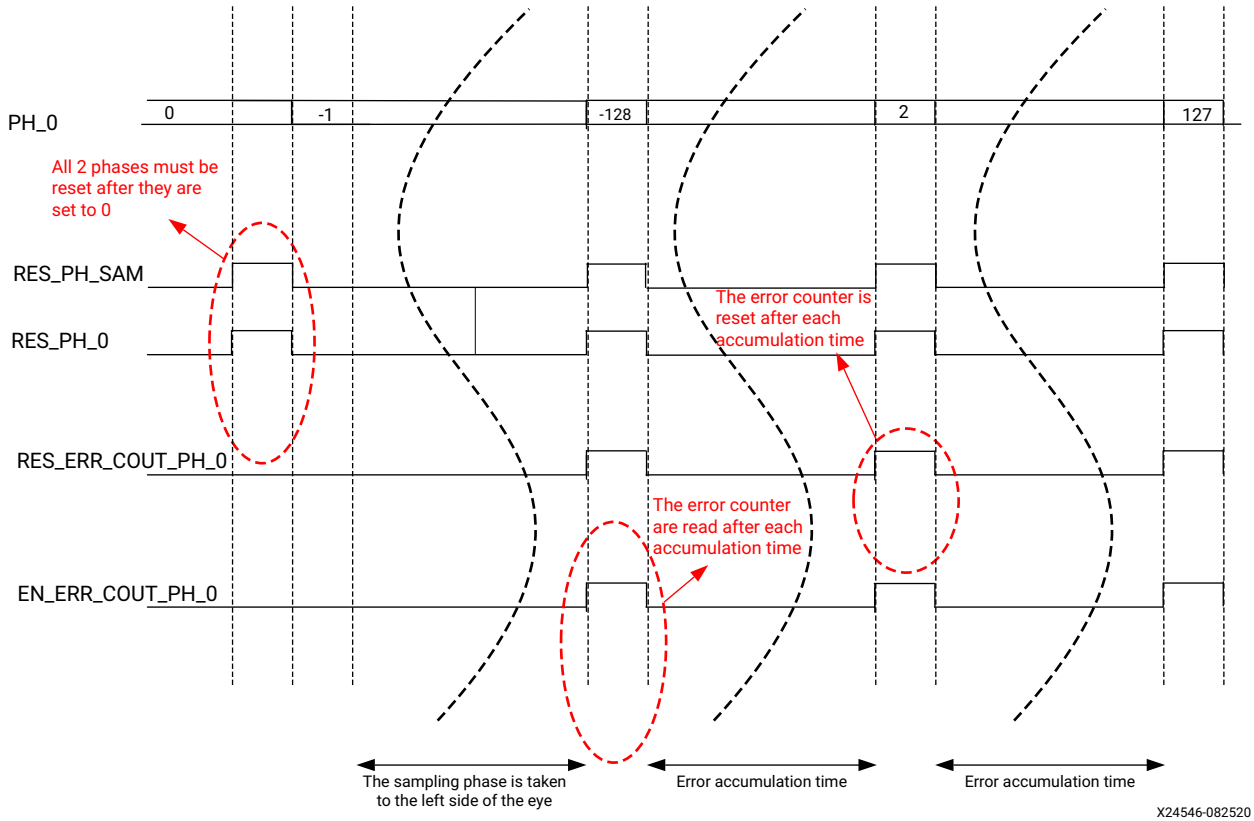


Figure 3: **Eye Scan Controller Timing Diagram with PH_NUM = 1**

Figure 4 shows the eye scan controller timing diagram in automatic mode when PH_NUM = 2.

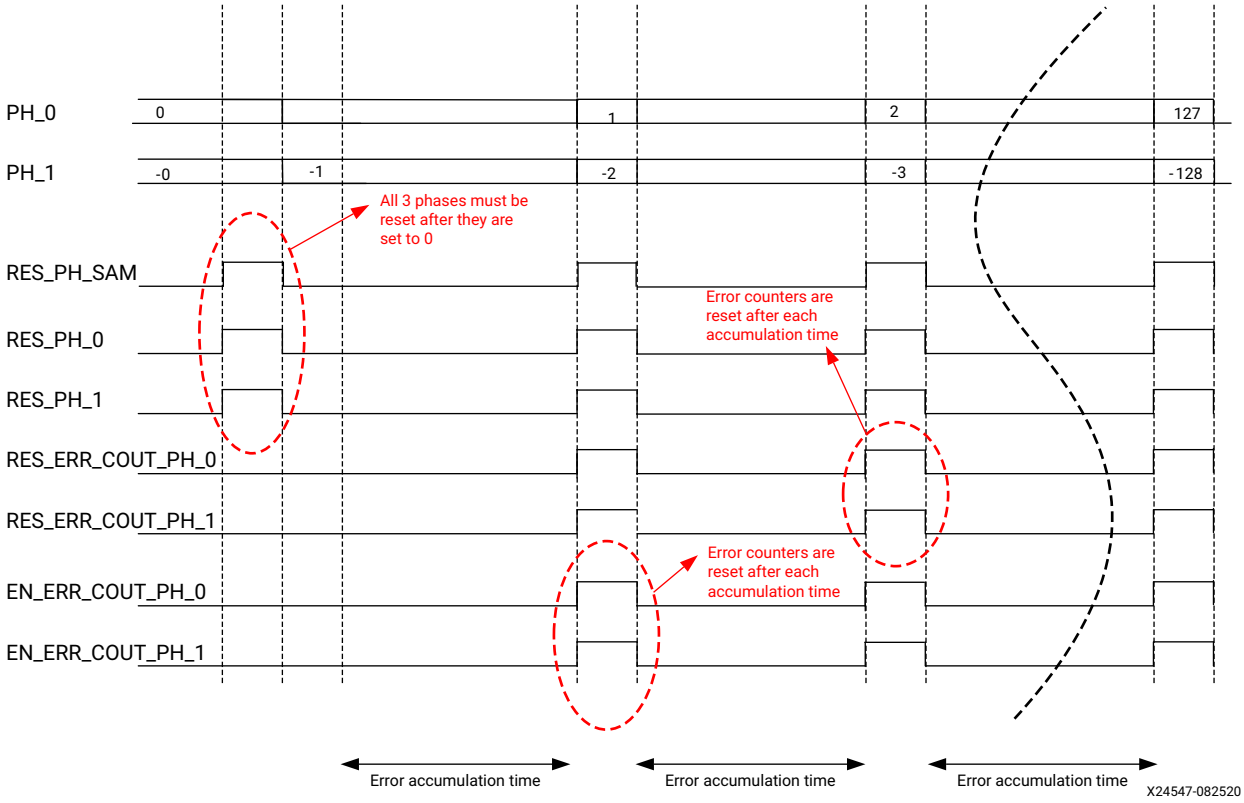


Figure 4: **Eye Scan Controller Timing Diagram in Automatic Mode with PH_NUM = 2**

The measurement time for each point is specified by the user through the port WAITING_TIME, which specifies the error accumulation time in clock cycles of F_{REFCLK}. Equation 7 allows relating the value of WAITING_TIME to the target BER:

$$WAITINGTIME = \frac{O\_R}{DTINWIDTH \cdot BER} \qquad \text{Equation 7}$$

To plot the eye scan, plot the ports ERR_PH_0 and ERR_PH_1 in the simulator viewer on an ILA core. ERR_PH_0 and ERR_PH_1 signals are valid only when the corresponding EN_ERR_PH_x signals are set to 1.

When PH_NUM = 1, the full eye shape can be seen on one line. When PH_NUM = 2, the two halves of the eye are simultaneously plotted on two independent lines.

Example plots are shown in Simulating the NIDRU (Figure 7 and Figure 8) and FPGA Hardware Test Bench (Figure 13 and Figure 14).

When AUTOM = 1, the eyescan is acquired only once if TRIGGER_MODE is set to 0, otherwise it is acquired continuously. The port EYE_AP reports to the user the last measured eye aperture, and can be used easily as a continuous measurement of the link quality. The resolution is 1UI/256.

In manual mode (AUTOM = 0), both phases can be moved by the user to measure the eye aperture in any given point inside the eye. PH_0 is the port for phase 0, PH_1 is the port for phase 1.

IMPORTANT: *Never move the exploring phases by more than 1 step per clock cycle.*

The timing diagrams in automatic mode (implemented in the eye scan controller), can be used as examples to derive the timing diagram for any desired custom mode setup.

Sampling Point Shift

The eye-scan feature comes at a price in terms of resources (see Resources). To save hardware resources, it is possible to shift the NIDRU sampling point inside the eye to measure the eye aperture. With this method, the eye scan can be performed only during a debug session, because bit errors are visible as soon as the sampling phase is pushed closer to the borders of the eye.

The port SHIFT_S_PH can be used to move the sampling point. The port is specified in twos complement. By setting SHIFT_S_PH to 0 (default) the NIDRU samples the incoming data stream in the middle of the eye.

The eye scan ports operate independently on the port SHIFT_S_PH, in the sense that each eye scan is always performed relative to the middle of the eye diagram, and not to the setting of SHIFT_S_PH.

Simulating the NIDRU

The purpose of the TB_SIM_DRU_JITTER test bench is to simulate the ability of the NIDRU to operate at several popular data rates with synchronous and plesiochronous serial inputs. The simulation covers the six cases shown in Table 3 in sequence. The user application data rate can be added as an additional case.

Case 2 and case 4 show the ability of NIDRU to work at both fractional and integer oversampling rates. Case 6 shows the ability of the NIDRU to operate even at very low data rates, even 1 Kb/s, equivalent to a 125K oversampling rate.

Table 3: **Simulation cases**

Case Number	Protocol	Data Rate	Ref Clock	Oversampling Rate
1	Proprietary	250 Mb/s	125 MHz	10
2	OC3	155.52 Mb/s (+100 ppm)	125 MHz	16.075
3	SDI	270 Mb/s (+100 ppm)	148.5 MHz	11
4	OC3	155.52 Mb/s	155.52 MHz	20
5	OC12	622.08 Mb/s	125 MHz	4.019
6	Proprietary	1 Kb/s	125 MHz	2.5e6

To run the simulation script:

1. Open a DOS command window.
2. Change to the `scripts` directory.
3. Open ModelSim.
4. In ModelSim, run the script `run_sim_do`.

IMPORTANT: *Although the test bench has been tested with Modelsim only, the NIDRU core is also expected to operate with these simulation tools: Vivado® simulator, Mentor Graphics Modelsim, and Synopsys VCS.*

The architecture implemented in the TB_SIM_DRU_JITTER test bench is shown in Figure 5. An ideal deserializer and serializer are used instead of a full SerDes to minimize the simulation time. The serializer and deserializer datapath is 20 bits or 32 bits, programmable through the DTIN_WIDTH attribute.

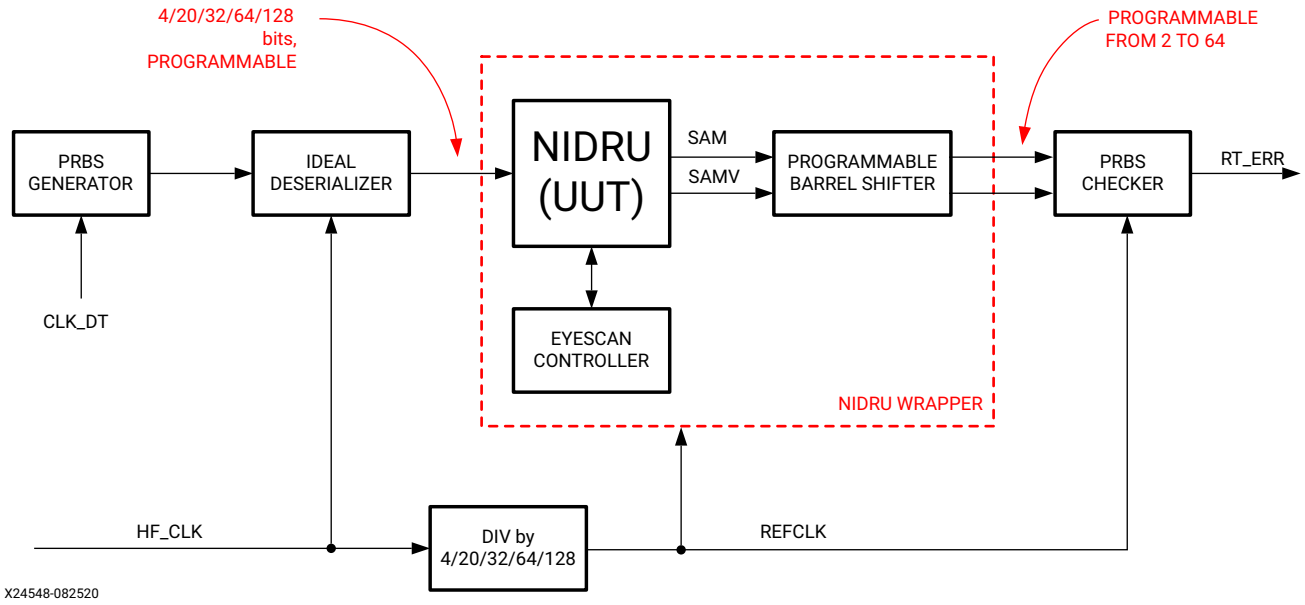


Figure 5: **TB_SIM_NIDRU Block Diagram**

The test bench contains two clock domains:

- The clock domain of the line, synchronized to CLK_DT
- The clock domain of the DRU, synchronized to REFCLK and to HF_CLK

The pseudo-random binary sequence (PRBS) generator works at full speed on CLK_DT and can generate any kind of industry standard PRBS [Ref 4]. The ideal deserializer, the NIDRU, and the PRBS checker all work on the local REFCLK domain.

During test case 1, a 1 ns step in the input datastream is applied (see Figure 6). The purpose of this is to show the ability of the NIDRU to respond exponentially to an input phase step.

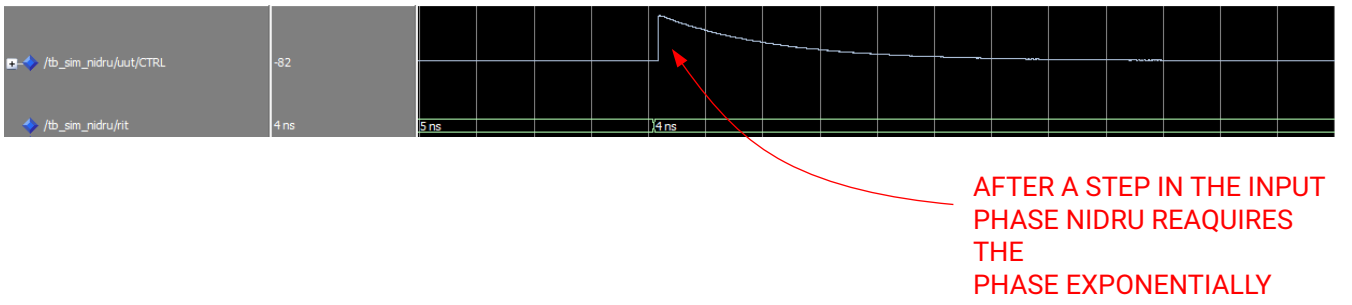


Figure 6: **NIDRU Response Following a 1 ns Step In the Input Phase**

During test case 2 and case 5, the eye scan is plot with PH_NUM = 1 (see Figure 7).

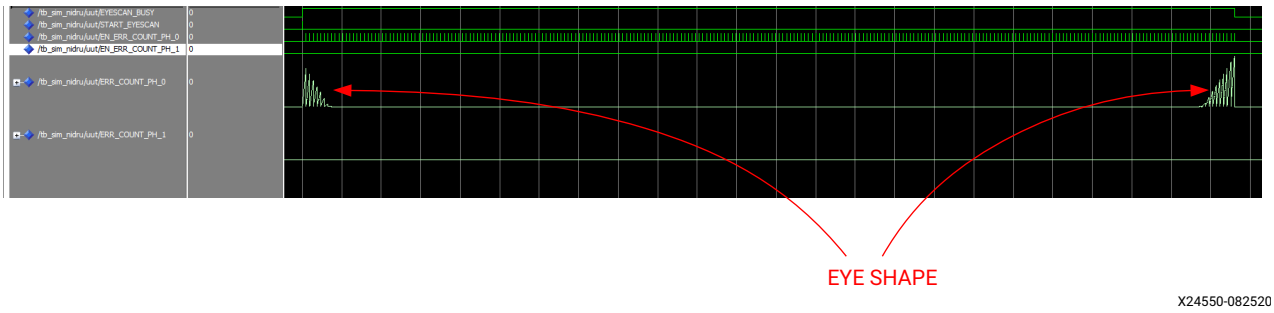


Figure 7: **Eye Scan with PH_NUM = 1 (This is a Simulation)**

An example of eye scan with two phases (PH_NUM = 2) is shown in Figure 8.

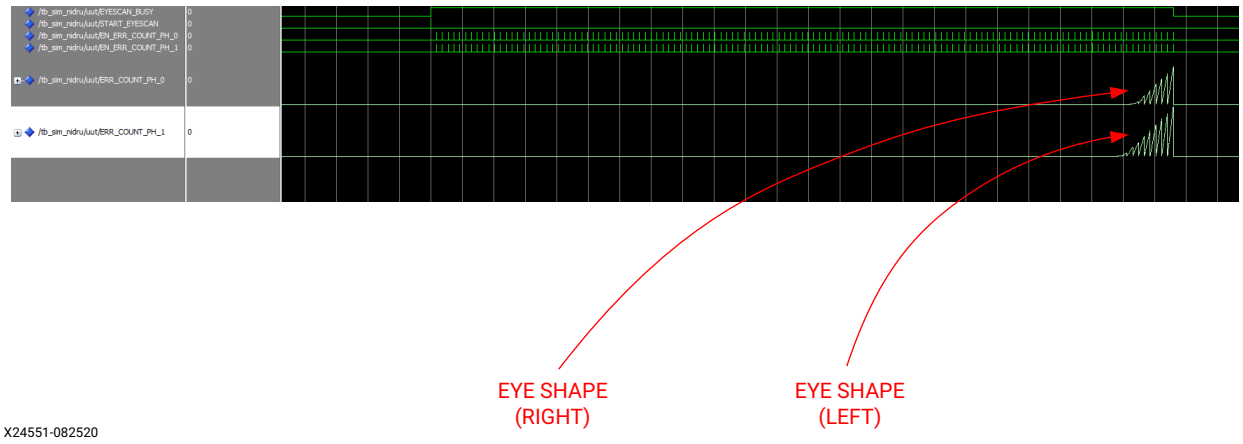


Figure 8: **Eye Scan with PH_NUM = 2 (This is a Simulation)**

FPGA Hardware Test Bench

Test bench TB_HW_ZUP is available as part of the reference design. The test bench TB_HW_ZUP is designed for the ZCU102 demonstration board. This test bench can be implemented to show the NIDRU data recovery capability with both synchronous and asynchronous inputs.

To compile the test benches, source the script `nidru_design_zup.tcl` from the Vivado Design Suite. The test bench architecture is shown in Figure 9.

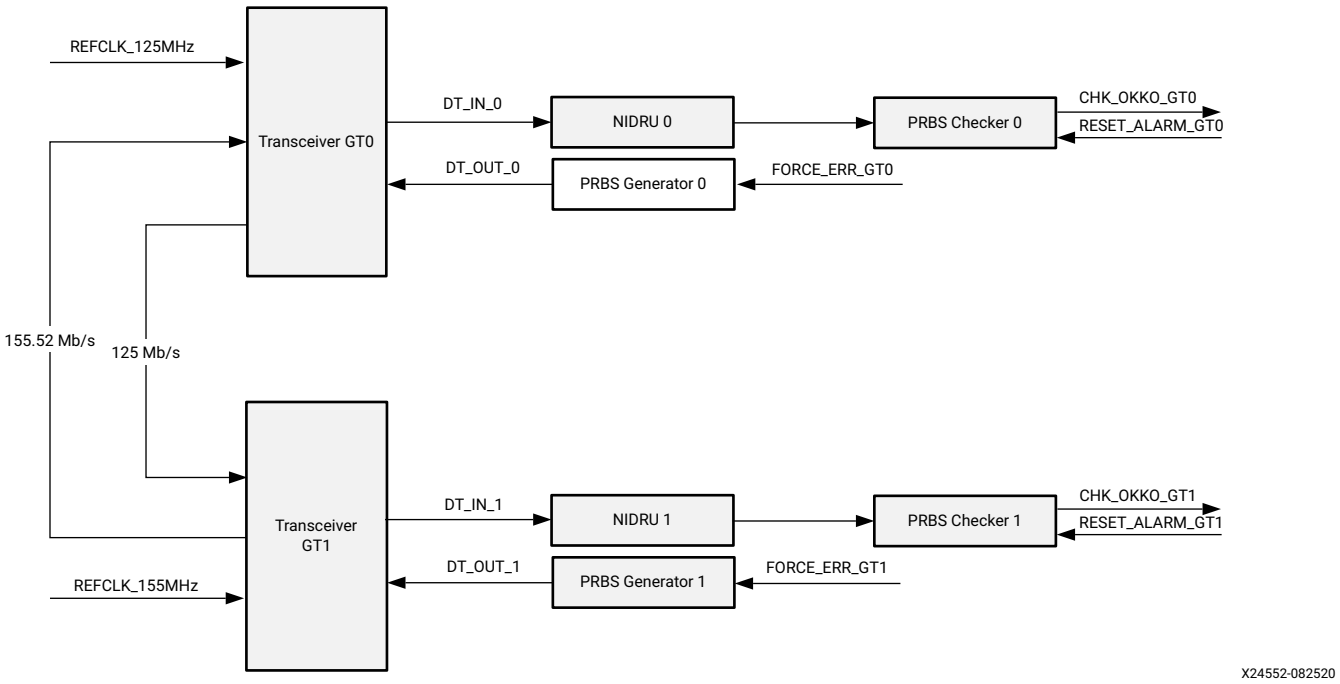


Figure 9: **TB_HW_DRU Test Bench Architecture**

The test bench includes:

- Two STM1/OC3 receivers, based on NIDRU, operating on a 125 MHz reference clock.
- Two Fast Ethernet receivers, based on NIDRU, operating on a 155.52 MHz reference clock.

Channel 0 transmits data synchronized with REFCLK 125 MHz, while Channel 1 transmits data synchronized with REFCLK 155.52 MHz. The GTHE4 Channels are cross connected via a SFP cable so that each receiver receives data at a frequency not synchronized to its own reference clock.

The two reference clocks are generated on board by using the system controller. The reference clock frequencies are configured through the serial interface on the ZCU102 board.

In each Quad, only two SerDes are used.

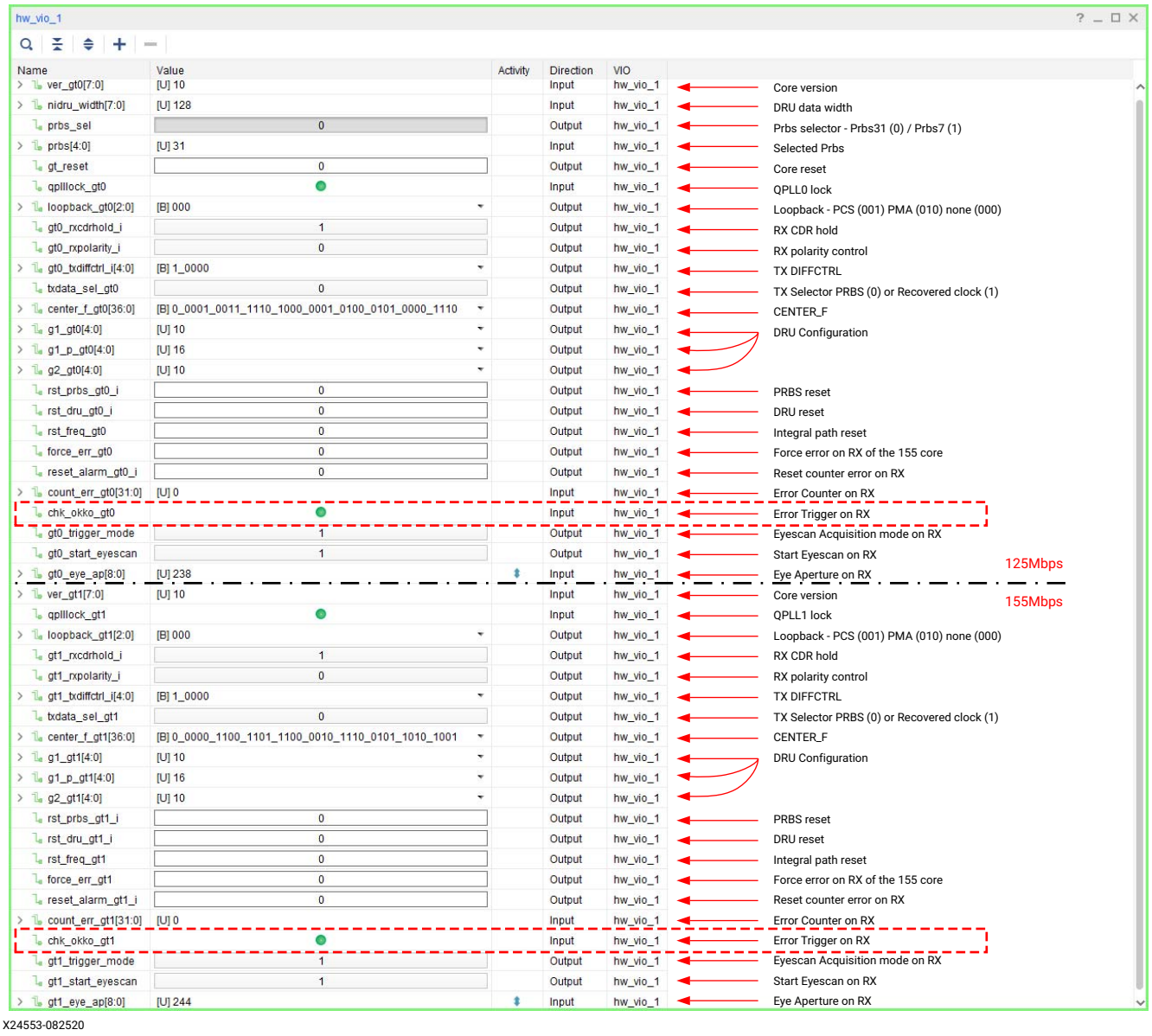
Each of the two channels is equipped with:

- A PRBS generator continuously sending a PRBS 7 or PRBS 31 pattern. The user can force each of the two PRBS generators to generate an error using the Vivado Logic Analyzer to show error detection on the corresponding PRBS checker.
- A PRBS checker continuously checking the incoming PRBS 7 or PRBS 31 pattern. The ERR output indicates detection of at least one error from the last ERR_RST. ERR is connected to the virtual input/output (VIO) and checked in real time. An error counter is also provided.

The specific PRBS pattern used in this application note for both the generator and the checker is based on the polynomial $x^{31} + x^{28} + 1$ for PRBS 31, $x^7 + x^6 + 1$ for PRBS 7 and can be changed to any other industry standard PRBS type.

Each PRBS checker works on the data delivered by the barrel shifter, which is instantiated right after each NIDRU block. Figure 10 reports the detailed description of all signals of the test bench which are controlled by the Vivado Logic Analyzer.

Both the 125 and 155 blocks are controlled in the same way, but with a different VIO. The pin names are consistent across the VHDL code, the logic analyzer project, and this application note.



X24553-082520

Figure 10: **Vivado Logic Analyzer Controlling the Demonstration Test Bench**

Each transmitter has the option to be set to generate a PRBS pattern, as described previously, or to synthesize a recovered clock. This mode, which can be activated on the fly, allows showing the capability of NIDRU to synthesize the recovered clock.

When the application works properly, all LEDs are green.

In case of an error in the datapath, the corresponding LEDs (highlighted with dashes in Figure 10) for the signals `chk_okko_gt0` and/or `chk_okko_gt1` are red.

The example design needs two asynchronous and independent clocks (155.52 MHz and 125 MHz). The ZCU102 board can provide two different clocks through the system controller. In the example design supplied with this application note, the system controller can be programmed by the System Controller User Interface and using the serial interface on the ZCU102 board.

The System Controller GUI software rdf0382-zcu102-system-controller-c-2019-1 can be downloaded from the ZCU102 Product Page [Ref 3].

Figure 11 shows the setup to generate the correct frequencies of 125 MHz and 155.52 MHz.

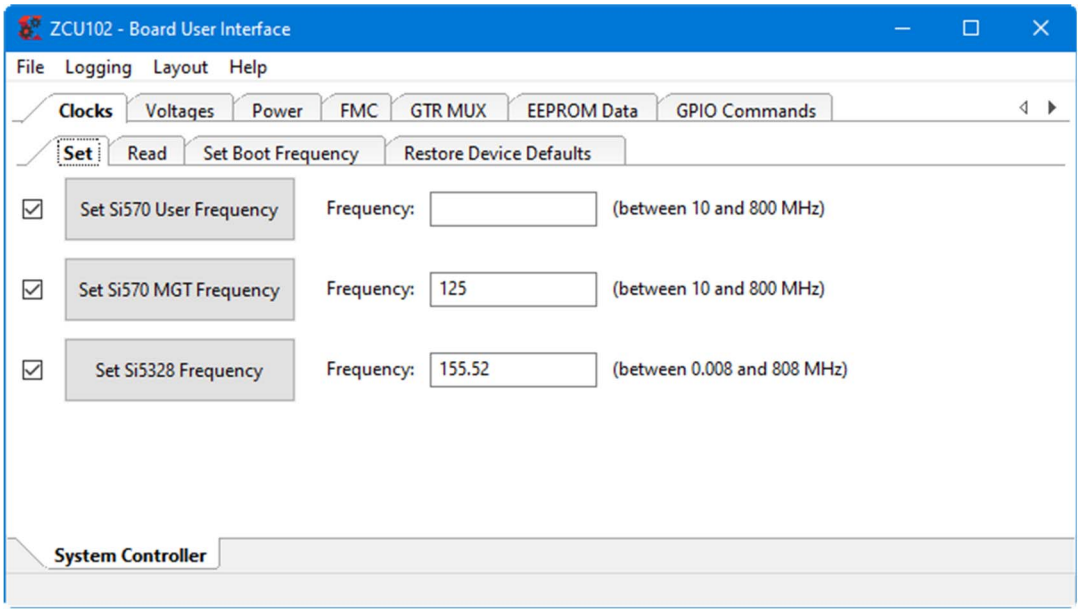


Figure 11: **Vivado Logic Analyzer VIO Controlling the SuperClock-2 Module**

To test the eye-scan feature, configure the ILA **Capture Mode Settings** and **Trigger Mode Settings** as shown in the Figure 12.

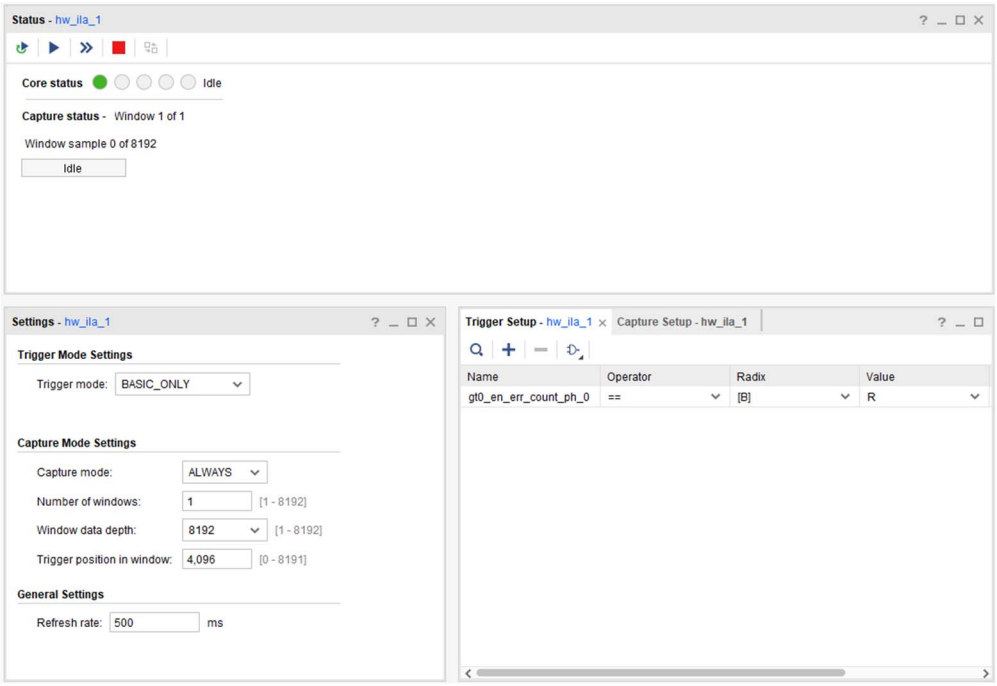


Figure 12: **Vivado Logic Analyzer ILA Settings**

The trigger is done on the rising edge of the signal EN_ERR_COUNT_0 asserting the signal START_EYESCAN available in the VIO window.

Figure 13 and Figure 14 show a hardware eye scan in case of PH_NUM = 1 and PH_NUM = 2, respectively.

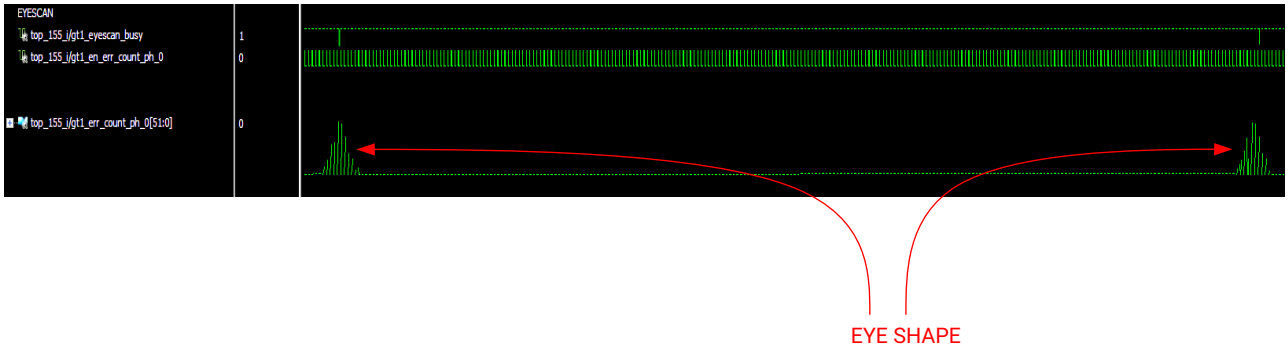


Figure 13: **Eye Scan with PH_NUM = 1 (This is a Hardware Measurement)**

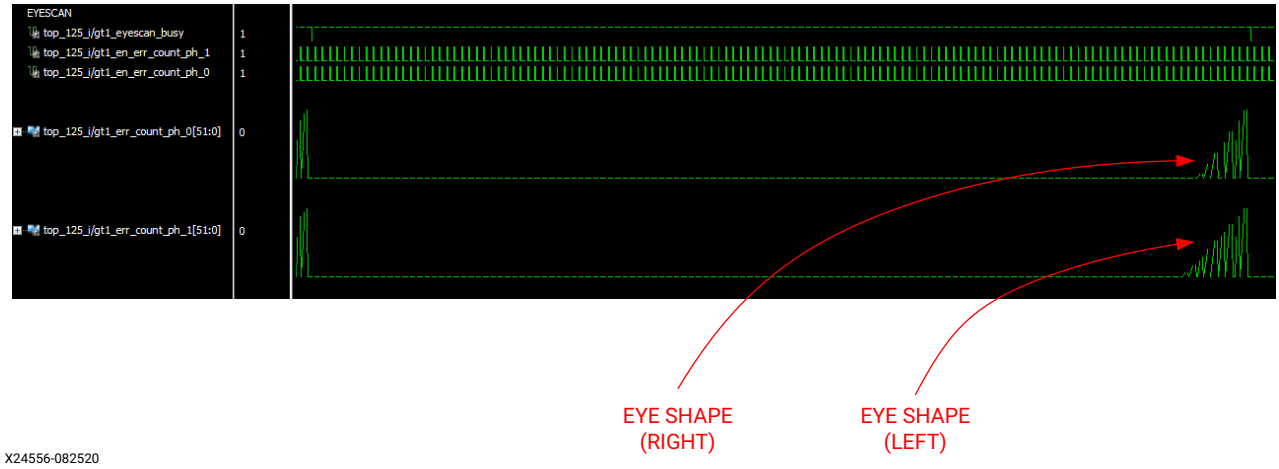


Figure 14: **Eye Scan with PH_NUM = 2 (This is a Hardware Measurement)**

PHY Configuration

This section provides recommendations for correctly configuring the PHY.

The NIDRU processes oversampled data from a PHY, which is usually a SelectIO interface or a SerDes. In the case of a SerDes, it has to be configured in lock to reference mode, and its auto-adapting equalizer should be disabled by setting these ports to the values listed here:

- RXCDRHOLD = 1
- RXLPMEN = 1
- RXLPMHFOVRDEN = 1
- RXLPMLFKLOVRDEN = 1

- RXOSOVRDEN = 1

These ports are available in the test bench through VIO. The following section describes configuring the GTH transceiver using the UltraScale FPGA Transceiver Wizard in the IP Catalog.

RECOMMENDED: *Download the most recent version of the IP core before using the wizard. For details on how to use this wizard, see the UltraScale FPGAs Transceivers Wizard: LogiCORE IP Product Guide (PG182) [Ref 5].*

Configure the **Basic** tab as shown in Figure 15. Set the receiver line rate at the oversampling rate. In the example, the line rate is set to 2.5 Gb/s which is 20 times the Fast Ethernet rate. The transmitter can be used to synthesize the recovered clock by setting it at the same rate as the receiver.

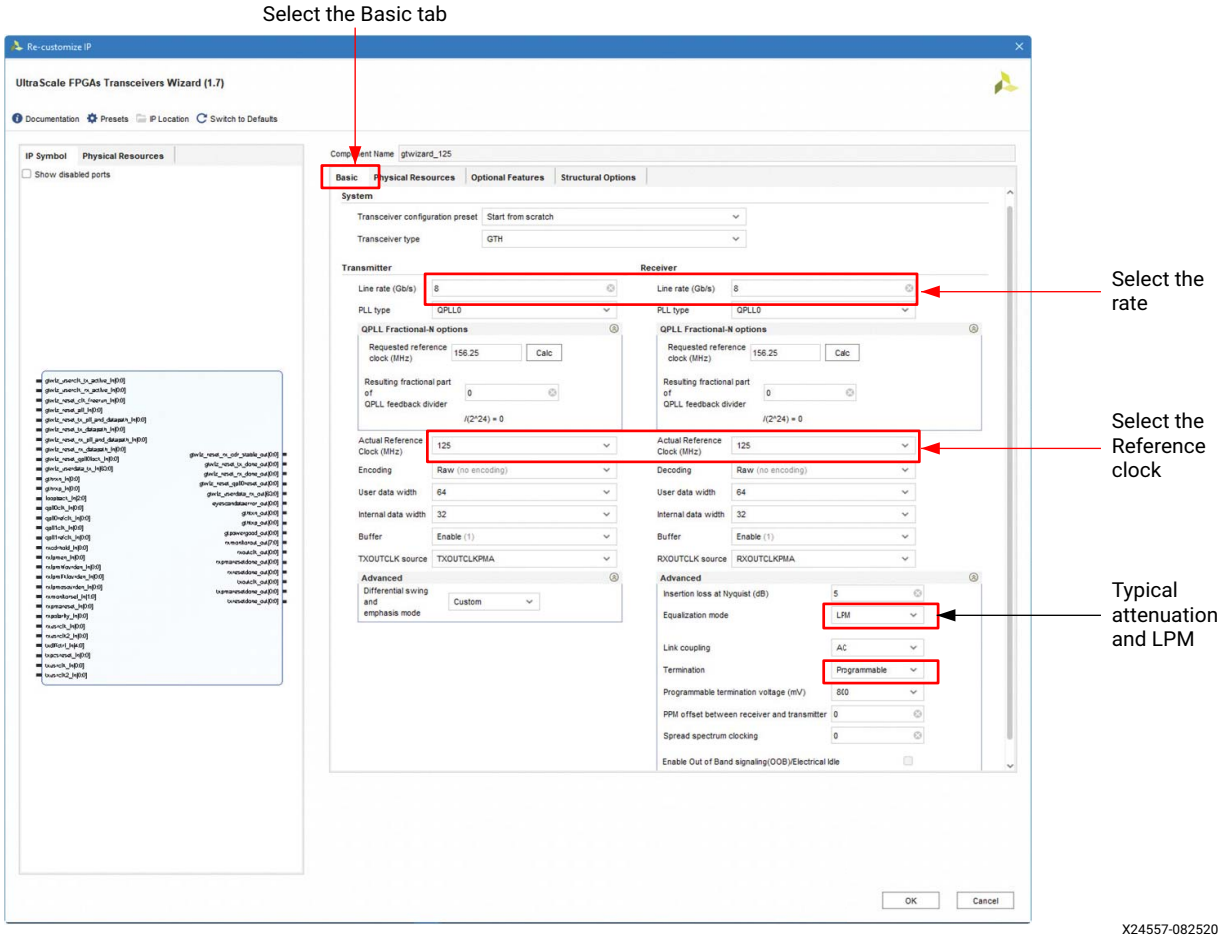


Figure 15: **Configuration of the Basic Tab**

Under the **Structural Options** tab, check all the ports that are highlighted in Figure 16 and Figure 17 to expose them in the generated wrapper. All of the ports chosen to be exposed must be set to 1.

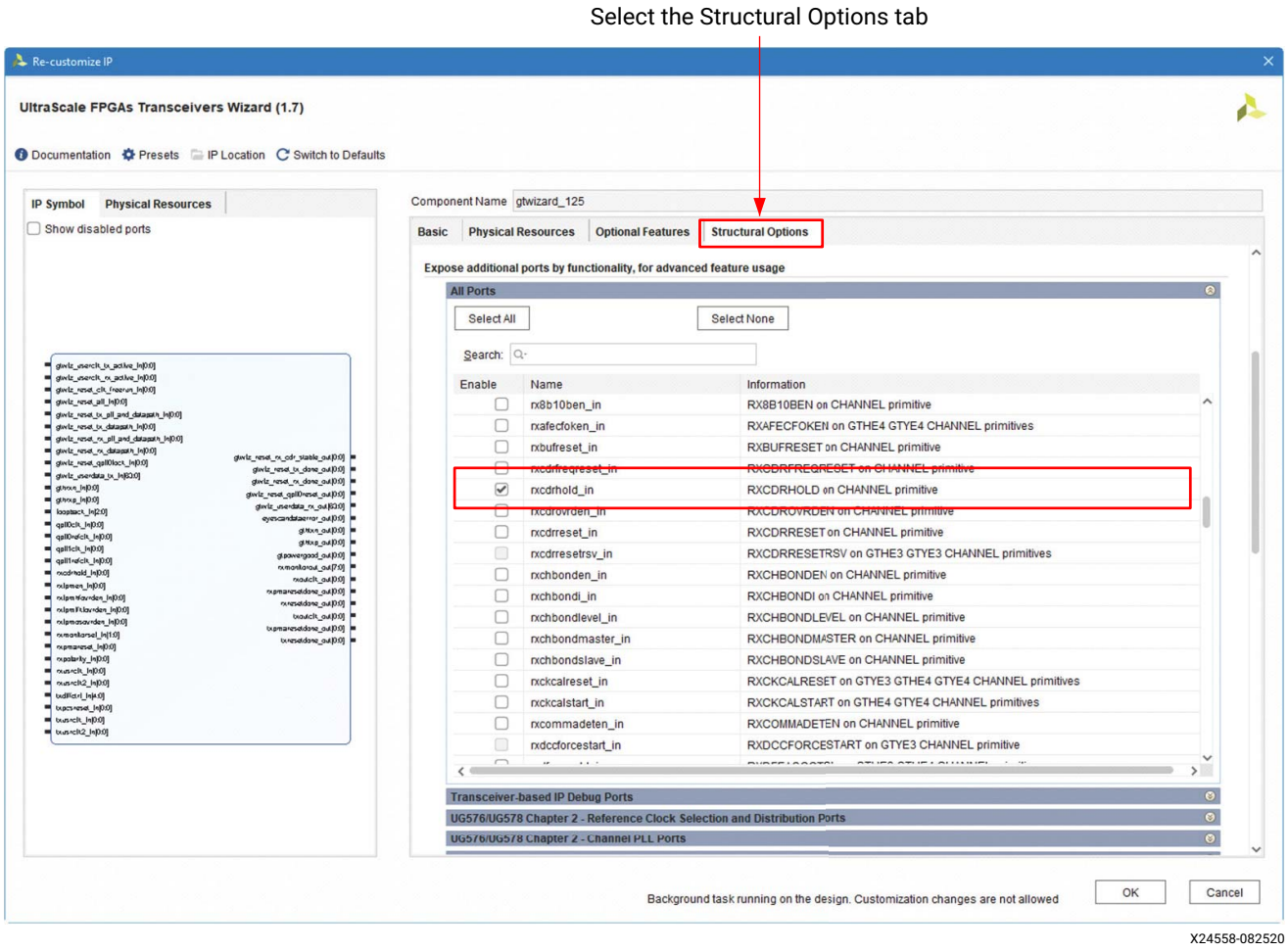


Figure 16: **Ports to be Exposed in the Structural Options Tab**

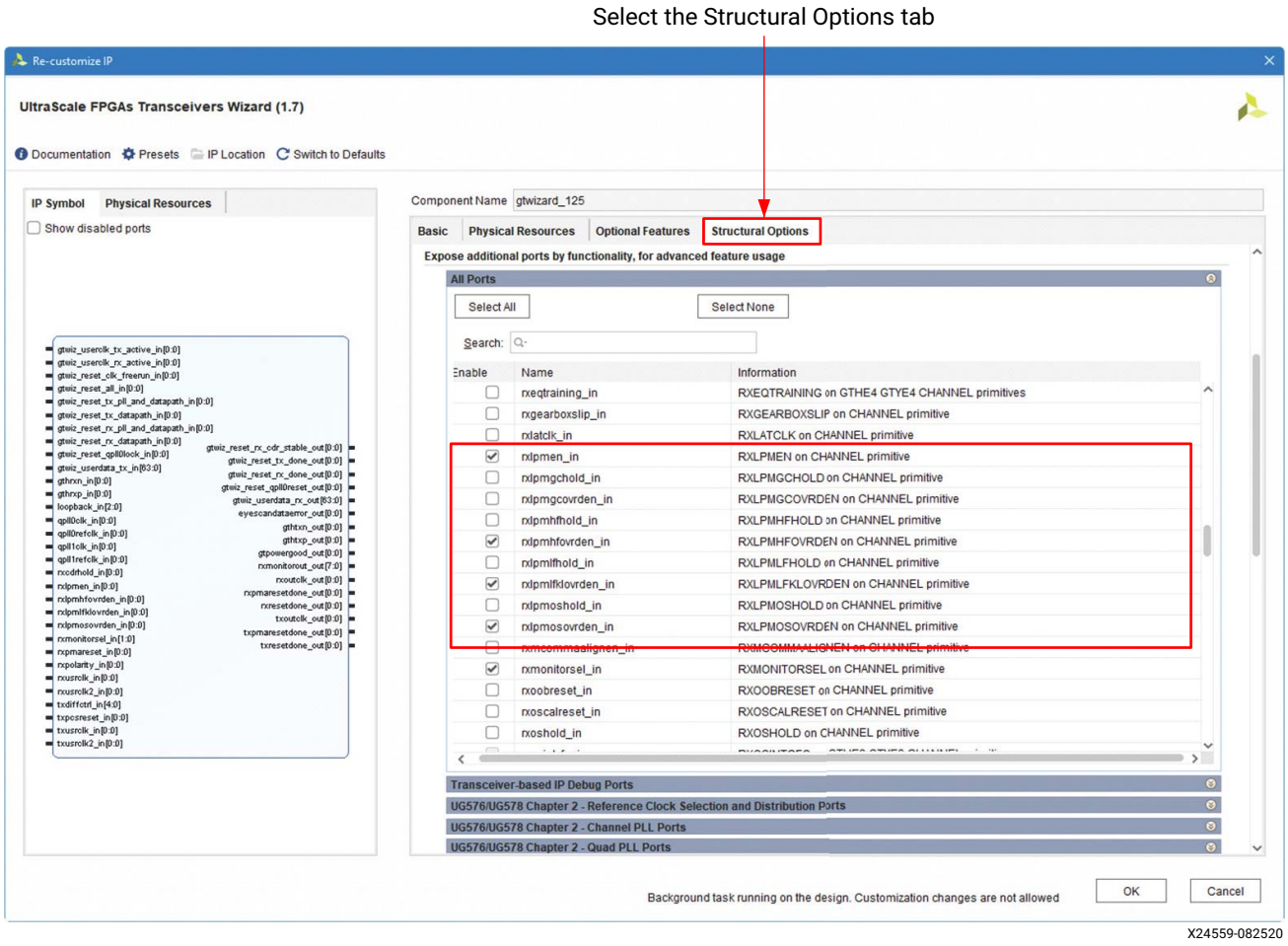


Figure 17: **Additional Ports to be Exposed in the Structural Options Tab**

Resources

The NIDRU is designed with efficient structures only (adders, multipliers, accumulators, and shifters). The resource requirements for UltraScale+ FPGAs are summarized in Table 4.

Table 4: **Hardware Resources Required for the NIDRU in UltraScale+ FPGAs**

Synthesis Type	Eye Scan (PH_NUM)	Flip-Flops	LUTs	BUFGs(1)
4	0	251	300	1
	1	556	585	
	2	697	689	
20	0	308	356	1
	1	631	715	
	2	781	860	

Table 4: **Hardware Resources Required for the NIDRU in UltraScale+ FPGAs *(Cont'd)***

Synthesis Type	Eye Scan (PH_NUM)	Flip-Flops	LUTs	BUFGs[1]
32	0	973	1,178	1
	1	1,846	2,363	
	2	2,271	2,906	
64	0	1,908	2,526	1
	1	3,802	5,449	
	2	4,743	6,863	
128	0	3,484	5,187	1
	1	7,185	11,257	
	2	9,031	14,223	

Notes:
1. Only one BUFG is required even if many channels are being set up, and even if all are working at different data rates.
2. These results were obtained using Vivado Design Suite, version 2016.4. The strategy used for the synthesis and implementation was Default and the CLK period was 6.4 ns.

Software Requirements

The software required for the design is listed here:

- Vivado Design Suite, version 2022.1 or later.
- Mentor Graphics ModelSim software, version 10.6b or later (for simulation).

Reference Design

Download the reference design files for this application note from the Xilinx website.

Table 5 shows the reference design matrix.

Table 5: **Reference Design Matrix**

Parameter	Description
General	
Developer name	Paolo Novellini, Antonello Di Fresco, and Giovanni Guasti
Target devices	7 series, UltraScale, UltraScale+, and Versal devices
Source code provided	Yes, partially encrypted
Source code format	VHDL
Design uses code and IP from existing Xilinx application notes and reference designs or third-party sources	This reference design uses code from the application note *An Attribute-Programmable PRBS Generator and Checker* (XAPP884) [Ref 4].

Table 5: **Reference Design Matrix *(Cont'd)***

Parameter	Description
Simulation	
Functional simulation performed	Yes
Timing simulation performed	N/A
Test bench used for functional and timing simulations	Yes
Test bench format	VHDL
Simulator software/version used	ModelSim 10.6b
SPICE/IBIS simulations	N/A
Implementation	
Synthesis software tools/versions used	Vivado synthesis
Implementation software tools/versions used	Vivado implementation
Static timing analysis performed	Yes
Hardware Verification	
Hardware verified	Yes
Hardware used for verification	ZCU102 board with Zynq UltraScale+ MPSoC

References

1. Best, Roland E. 1999. fourth edition. *Phase-Locked Loops: Design, Simulation, and Applications*. McGraw-Hill Professional Publishing
2. Gardner, Floyd M. 2005. third edition. *Phaselock Techniques*. Wiley-Interscience
3. ZCU102 Product Page (https://www.xilinx.com/products/boards-and-kits/zcu102)
4. *An Attribute-Programmable PRBS Generator and Checker* (XAPP884)
5. *UltraScale FPGAs Transceivers Wizard: LogiCORE IP Product Guide* (PG182)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/04/2022	3.1	Incorporates NIDRU v.11 with support for Versal devices. Added port types for TRIGGER_MODE and EYE_AP in Table 2. Updated Vivado Design Suite and Mentor Graphics ModelSim software versions in Software Requirements.
09/14/2020	3.0	Incorporates NIDRU v.10 which adds 64 and 128 bits datapath support. It also adds the ability to measure runtime eye aperture. Updated Reference Design for ZCU102 board.
02/28/2017	2.1	Incorporates NIDRU v.9 with 4 bits datapath support.

Date	Version	Revision
09/24/2015	2.0	Incorporates NIDRU v.8, with eye scan and support for 20 and 32 bits. Fixed a bug in the barrel shifter, preventing the user to use output amplitudes below 5. Title change, as the NIDRU v.8 has now a programmable input datapath.
04/17/2015	1.0	Initial Xilinx release.

Please Read: Important Legal Notices