XAPP1083 (v1.0.2) January 17, 2014

Fractional Burst Clock Data Recovery for XG-PON Applications

Author: Paolo Novellini and Massimo Chirico

Summary

This application note describes the implementation of an ITU.T G987-compliant fractional burst clock data recovery (BCDR) circuit for an optical line termination (OLT) operating in a 10 Gb-capable passive optical network (XG-PON) environment.

Introduction

XG-PON is ITU-T's next-generation optical access technology. One of the most challenging components in the XG-PON environment is the BCDR, operating on burst signals at 2.488 Gb/s.

Based on fully synchronous oversampling technology, the implementation of the BCDR described in this application note is well suited to the Kintex®-7 and Virtex®-7 FPGAs due to the low required minimum oversampling ratio and the highly pipelined oversampling technology. The speed grade requirement is driven only by the GTX transceivers, which must run at 12.44 Gb/s.

Features

The BCDR circuit implementation described in this application note has these features:

- Fully synchronous design: Although the fractional nature of the BCDR core allows for the selection of a different reference clock for the TX and RX paths, the clock frequency can also be reused for the transmit path:
  - 80-bit datapath
  - Single clock at 155.52 MHz
- Fully fractional design: The ratio between the oversampling rate and the data rate can be an integer or a fraction. This implies that the core can operate at any rate and reference clock. The recommended operating condition is with a ratio of five. The other features of the design are:
  - On-the-fly programmable fractional level
  - Fractional burst acquisition
  - 1.244 Gb/s and 2.488 Gb/s burst operation
- Programmable preamble length up to 32 bits: The preamble length identifies the minimum number of consecutive `01` bits to flag a preamble. For long preambles, multiple and consecutive preambles are flagged by the BCDR. It is recommended to keep these at 32.
- Hitless programmable bandwidth during tracking: This core is able to track jitter during the payload, i.e., outside of the burst area. To optimize robustness, the bandwidth is digitally user-adjustable even at runtime.
- Programmable averaging level during burst acquisition of 16, 32, 64, or 128 bits: The statistical information in consecutive 32-bit preambles can be used to increase the accuracy of phase estimation during the burst. It is recommended to keep the number of bits lower than the preamble length specified by the OLT.

XG-PON Network Overview

This section is optional for the experienced reader. Its main purpose is to outline the operating principle of an XG-PON access network, particularly from the topology point of view.

Figure 1 illustrates the XG-PON architecture for downstream transmission. The OLT transmits a single optical stream at 9.95 Gb/s to the passive splitter. The splitter replicates the data stream to each optical network terminal (ONT) connected to it. The downstream data transmission is continuous, thus all ONTs do not operate in burst. The data received by all ONTs is the same, but only a fraction of that known as a slot can be decoded by each ONT.

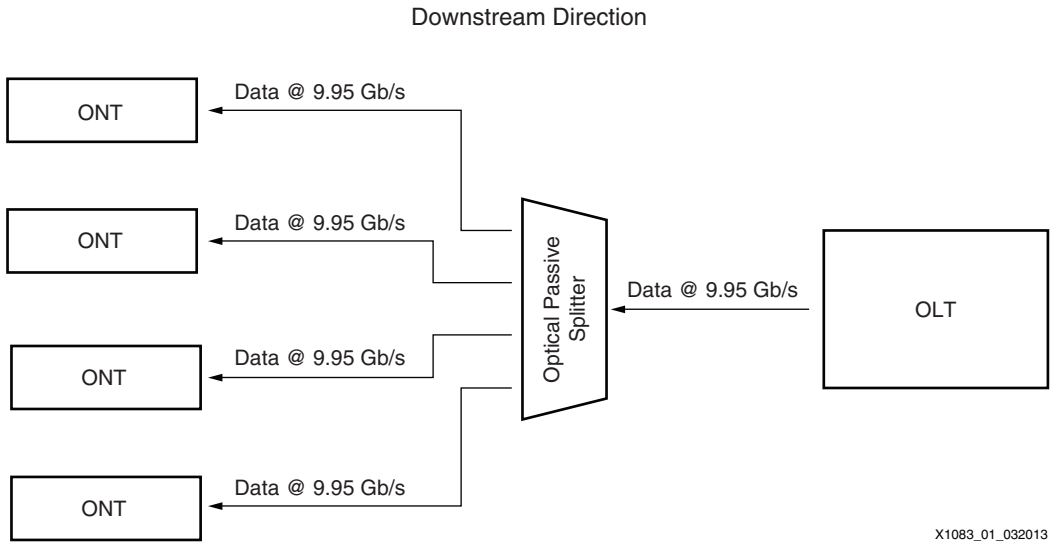

Figure 1: **XG-PON Architecture for Downstream Transmission**

Figure 2 shows how each ONT recovers the clock embedded into the received data, cleans it up, and reuses it to clock the upstream transmission. The raw upstream speed is 2.488 Gb/s. Each ONT transmits data at the same frequency. However, data from different ONTs arrives at the OLT at a phase that is completely uncontrolled and varies significantly over time and temperature. To avoid collision, each ONT must send data only during its permitted time slot. The time sharing across ONTs is controlled by the OLT medium access control (MAC) layer.

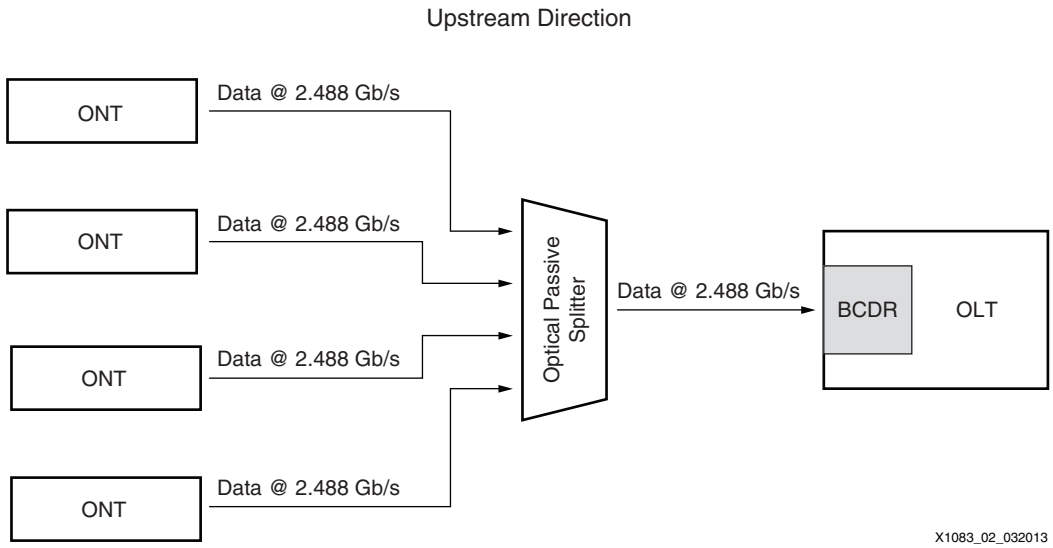

Figure 2: **XG-PON Architecture for Upstream Transmission**

When a new ONT is allowed to send data to the OLT, the BCDR acquires its phase and extracts the raw data in each burst. Each burst allocates adequate time for the BCDR to:

- Acquire the sampling phase.
- Identify a start- and end-of-packet to identify the packet boundary.
- Allow guard time for ONTs to power on and off their laser sources.
- Allow the automatic gain equalizer in the OLT to settle.

All these contributions affect the efficiency of the upstream direction. The downstream direction is a continuous transmission and is thus much more efficient compared to the upstream direction. This architectural limitation fits very well with the application requirement, however, because users generally require more bandwidth in the downstream direction rather than upstream. Figure 3 shows the data flow in both the downstream and upstream directions. It highlights the preamble, which is only required for upstream transmission. The preamble is a periodic repetition of the `10` bit pattern. This pattern allows maximizing the statistical information in the preamble to optimize the overall upstream efficiency. The length of the pattern is set by the OLT to a value that allows its BCDR to acquire the burst phase. The bottom part of Figure 3 shows an example configuration of phases.

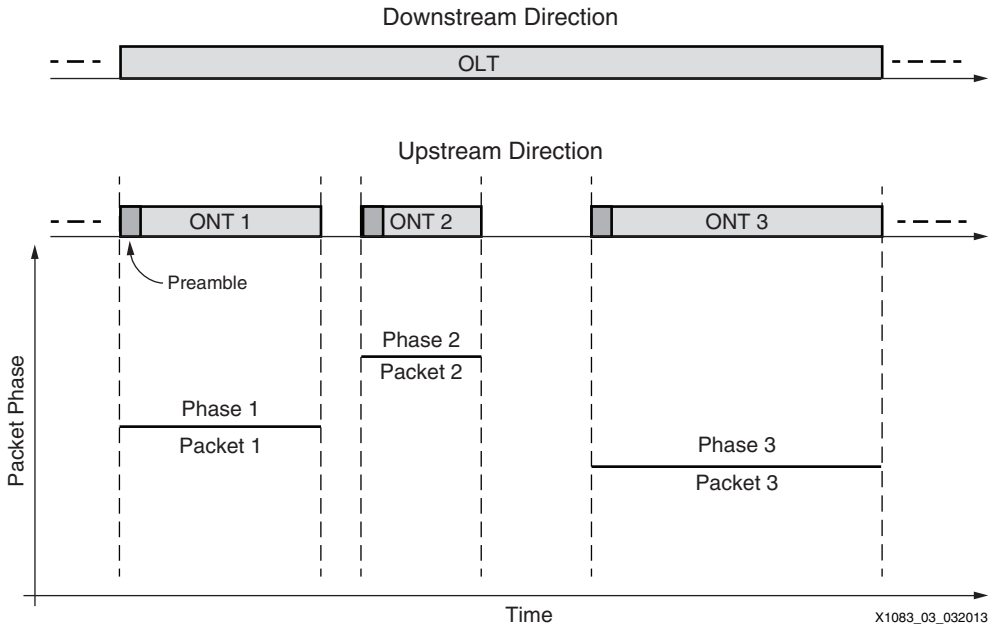

Figure 3: **XG-PON Architecture for Upstream and Downstream Transmissions**

Note: Although the BCDR in this application note can work with any preamble length, it is recommended to keep the preamble to a length of at least 32 bits to provide adequate phase information during burst phase acquisition.

Circuit Description

Figure 4 illustrates the BCDR architecture showing the relevant inputs and outputs. The 80-bit wide deserialized data flows in parallel into the lower branch and upper branch, respectively.

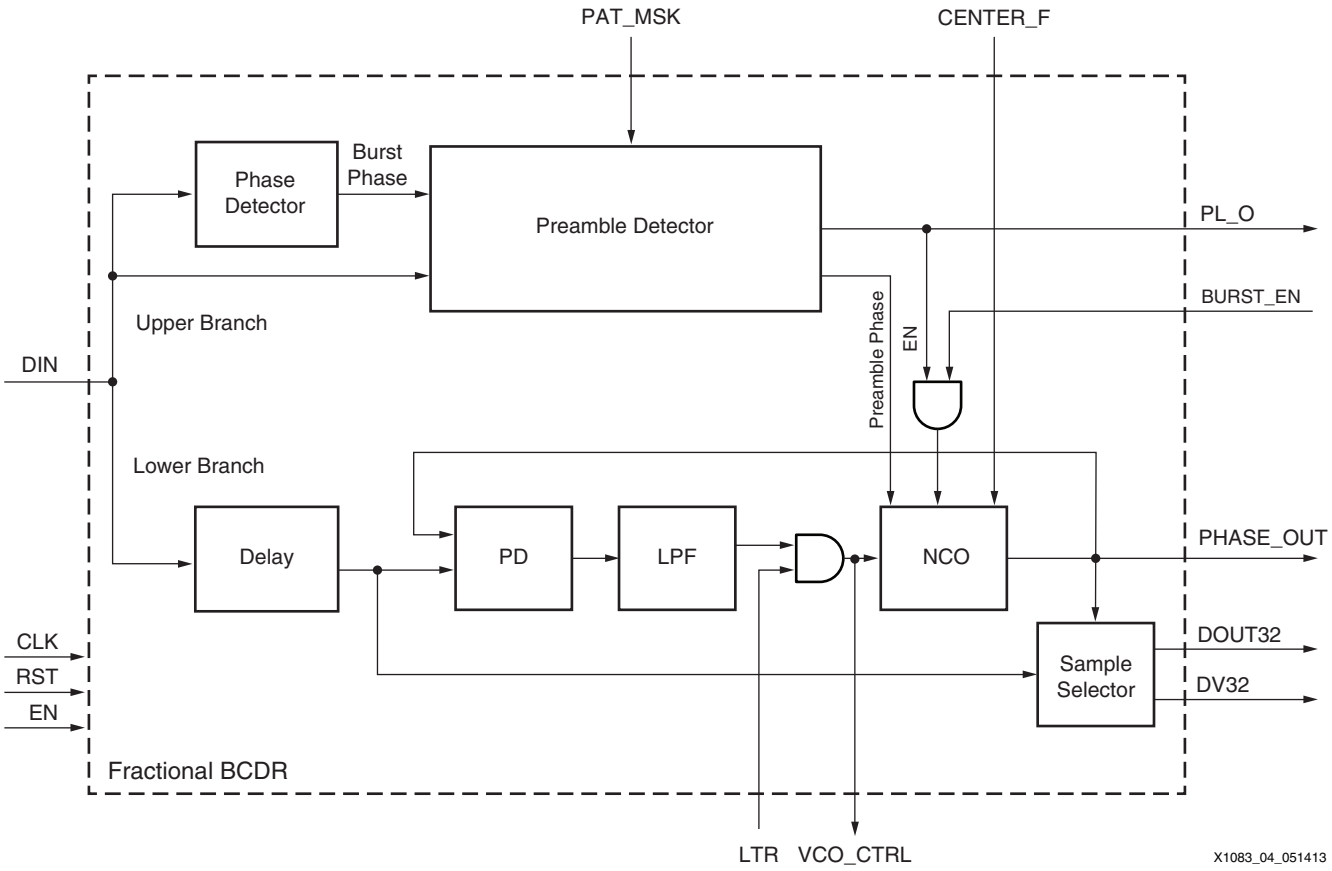

Figure 4: **Fractional BCDR Architecture Simplified Diagram**

The lower branch works on delayed data, continuously tuning the numerically controlled oscillator (NCO) to track the incoming data edges. Each raw sample is associated with a phase between –180 degrees and +180 degrees. The raw sample with the phase closest to 0 degrees (i.e., closest to the middle of the eye diagram) is extracted by the sample selector block. The lower branch tracks phase variations with typical time constants that are much longer than the preamble time. Thus, the loop in the lower branch is expected to track phase changes or jitter, but not bursts.

The delay element allows the upper branch to recognize a preamble and estimate its phase by averaging the phase information of many consecutive edges. Upon recognizing the consecutive edges as a preamble, the NCO in the lower branch is steered in one single clock cycle to be aligned with the new packet. The lower branch never experiences a phase burst because the NCO is steered just before the burst enters the phase detector in the lower branch.

For debugging purposes, it is possible to disable the burst injection capability by setting BURST_EN to 0. BCDR Simulation Test Bench, page 7 describes this item, which is the characterizing feature of a BCDR. The phase of the NCO can be monitored over time for debug purposes, both in simulation and in hardware, via the signal PHASE_OUT (15 downto 0).

BCDR Core Ports

Table 1 describes the ports of the BCDR core.

Table 1: **BCDR Core Ports Description**

Port	Type	Default	Description	Comment
CLK	IN std_logic	N/A	Clock	Requires a 155.52 MHz refclk, typically from the SerDes.
RST	IN std_logic	1	Reset	Active Low.
EN	IN std_logic	1	Enable pin	Connected to all internal processes of the BCDR. Set to 1.
DIN	IN std_logic_vector(79 downto 0)	N/A	Input data	MSb is conventionally the latest bit coming in. This is the same convention used in the 7 series SerDes.
DOUT32	std_logic_vector(31 downto 0)	N/A	Output data	MSb is conventionally the latest bit coming in. Data is grouped in 32 bits.
DV32	OUT std_logic	N/A	Output data valid	When High, DOUT32 is valid.
CENTER_F	IN std_logic_vector(36 downto 0)	10000000 00000000 00000000 00000000 00000	Center frequency	Sets the fractional ratio between the oversampling data rate and incoming data rate. By default, CENTER_F must be set to h1000000000.
PAT_MSK	IN std_logic_vector(5 downto 0)	100000	Pattern mask	Unsigned decimal, default is 32. Specifies to the BCDR how many bits to detect in the preamble.
BURST_EN	IN std_logic	1	Activate preamble detection	Debug signal, default set to 1. When set to 0, the burst is detected, but the NCO phase is not steered accordingly.
BDW	IN std_logic_vector(4 downto 0)	1010	Tracking bandwidth	Set to 01010. Each step down doubles the CDR bandwidth when tracking.
PL_O	OUT std_logic	N/A	Burst detected	Debug signal, indicates that a preamble has been detected.
AVE_SEL	IN std_logic_vector(1 downto 0)	0	Averaging level	This is an unsigned integer that indicates how many preamble bits should be used to estimate the starting phase of the packet: 0: 16 1: 32 2: 64 3: 128 The default is 0.

Table 1: **BCDR Core Ports Description *(Cont'd)***

Port	Type	Default	Description	Comment
PHE_BST_DRU	OUT std_logic_vector(15 downto 0)	N/A		Phase detector error provided by the lower branch in the BCDR.
PHE_BST_BURST	OUT std_logic_vector(15 downto 0)	N/A		Debug signal. Phase profile of incoming packets as seen by the BCDR.
PHE_BST_BURST_AVE	OUT std_logic_vector(15 downto 0)	N/A		Debug signal. Average of PHE_BST_DRU and PHE_BST_BURST.
PHASE_OUT	OUT std_logic_vector(20 downto 0)	N/A	Output voltage-controlled oscillator (VCO) phase	Debug signal. The current NCO phase can be read at each clock cycle. The phase is over the 16 least significant bits in signed format.
LTR	IN std_logic	0	Lock to reference	Debug signal. When set to 1, the BCDR no longer tracks data. The burst detection is still controlled by BURST_EN.
VCOCTRL	OUT std_logic_vector(31 downto 0)	N/A	VCO control	Debug signal. The control signal of the NCO.
RECCLK	OUT std_logic_vector(79 downto 0)	N/A	Bursty recovered clock	Debug signal. Can be serialized by a 12.44 Gb/s SerDes to synthesize the bursty recovered clock.

The deserializer data enters from DIN and exits from DOUT32. The output DOUT32 is valid only when DV32 is 1. CENTER_F specifies the ratio between the incoming datarate F_{DIN} and the reference clock F_{CLK}, and is calculated as follows:

$$\text{CENTER\_F} = \frac{F_{DIN}}{F_{CLK}} \cdot 2^{32} \qquad \textit{Equation 1}$$

In the example described in this application note, F_{DIN} is 2.488 Gb/s, F_{CLK} is 155.52 MHz, and CENTER_F yields `10000000000000000000000000000000` in binary format. This corresponds to an oversampling ratio of five. Five has been selected as the minimum oversampling ratio to guarantee robustness to optical module impairments. PL_O is a debug signal that the BCDR pulses to 1 each time it detects a preamble. If a preamble is detected, the NCO is steered to the estimated phase only if BURST_EN is set to 1, which is the default condition.

Table 2 shows the VHDL declaration and instance for the BCDR component.

Table 2: **VHDL Declaration and Instance for the BCDR Component**

```
COMPONENT GPON_80
PORT(
  CLK : IN STD_LOGIC;
  RST : IN STD_LOGIC;
  EN : IN STD_LOGIC;
  AVE_SEL : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
  DIN : IN STD_LOGIC_VECTOR(79 DOWNTO 0);
  CENTER_F : IN STD_LOGIC_VECTOR(36 DOWNTO 0);
  BDW : IN STD_LOGIC_VECTOR(4 DOWNTO 0);
  BURST_EN : IN STD_LOGIC;
  PAT_MSK : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
  LTR : IN STD_LOGIC;
  PL_O : OUT STD_LOGIC;
  DV32 : OUT STD_LOGIC;
  DOUT32 : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
  DOUT_BST_EN : OUT STD_LOGIC;
  DOUT_BST : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
  PHE_BST_DRU : OUT STD_LOGIC_VECTOR(15 DOWNTO 0);
  PHE_BST_BURST : OUT STD_LOGIC_VECTOR(15 DOWNTO 0);
  PHE_BST_BURST_AVE : OUT STD_LOGIC_VECTOR(15 DOWNTO 0);
  PHASE_OUT : OUT STD_LOGIC_VECTOR(20 DOWNTO 0);
  VCOCTRL : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
  RECCLK : OUT STD_LOGIC_VECTOR(79 DOWNTO 0)
);
END COMPONENT;
```

```
INST_GPON_80: GPON_80 PORT MAP(
  CLK => ,
  RST => ,
  EN => ,
  AVE_SEL => ,
  DIN => ,
  CENTER_F => ,
  BDW => ,
  BURST_EN => ,
  PL_O => ,
  PAT_MSK => ,
  DV32 => ,
  DOUT32 => ,
  DOUT_BST_EN => ,
  DOUT_BST => ,
  PHE_BST_DRU => ,
  PHE_BST_BURST => ,
  PHE_BST_BURST_AVE => ,
  PHASE_OUT => ,
  LTR => ,
  VCOCTRL => ,
  RECCLK =>

);
```

BCDR Simulation Test Bench

Setting Up the Simulation Environment

To run the BCDR simulation test bench:

1. Change the ModelSim working directory to the `simulation` folder.
2. Execute the `runsim.do` script from within ModelSim.
3. Map the work library in the script using the vmap command.
4. Refresh (command refresh).

The simulation script compiles all test bench files, runs the simulation, and configures the waveform viewer to show the simulation signals.

Simulation Test Results

Figure 5 illustrates the BCDR non-integer data recovery unit (NIDRU) test bench architecture.

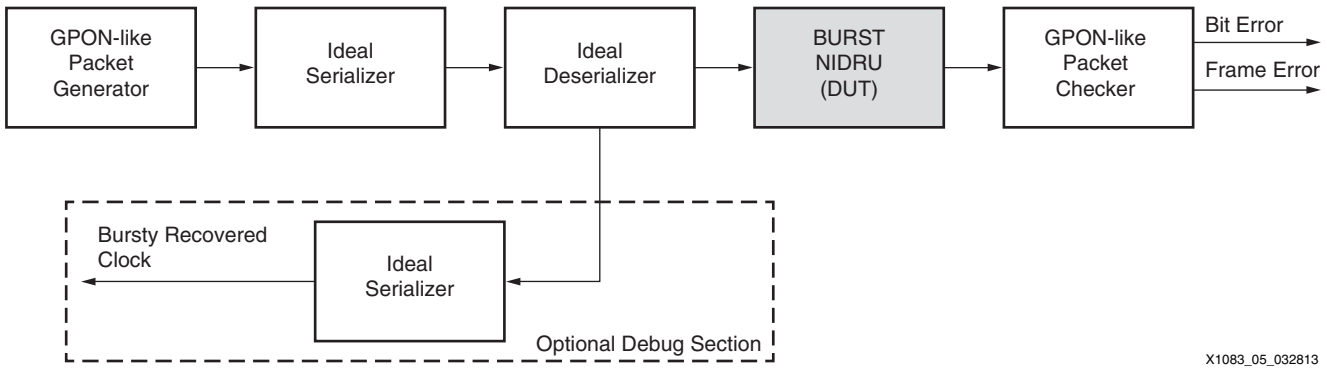

Figure 5: **Burst NIDRU Test Bench Architecture Block Diagram**

The simulation test bench block diagram includes:

- A G-PON-like pattern generator.
- An ideal serializer emulating the SerDes transmitter in the ONT.
- An ideal deserializer emulating the SerDes receiver in lock to reference mode.

- A G-PON-like pattern checker.
- An optional section that regenerates the bursty recovered clock for debugging.

Ideal serializers and deserializers are used instead of the real SerDes models to make the simulation platform independent and speed up simulation times.

Figure 6 shows the structure of the G-PON-like frame.

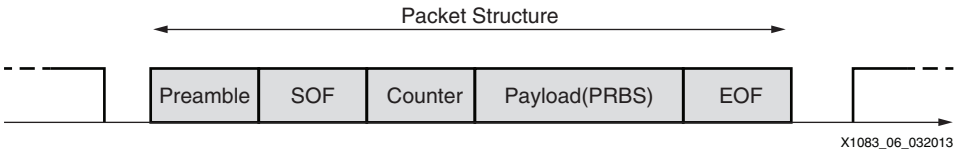

Figure 6: **G-PON-like Packet to Test the Burst NIDRU**

The G-PON-like pattern generator generates frames that start with a preamble and a start of frame (SOF) followed by a 16-bit counter that increases at each frame and wraps around. The payload is a truncated PRBS23 pattern.

The preamble has 48 bits with alternating 1 and 0 values. The pattern start is fixed at F628F628 and is 32 bits in length. Three bits are then set to 0 constantly and the following 16 bits are the counter. The zeros are implemented to prevent the counter emulating the start of the pattern.

Note: The pattern start value of F628F628 has been conventionally chosen from the synchronous digital hierarchy (SDH) world.

Any bit error is detected by the truncated PRBS23 error checker. When the output dt_burst_check in the test bench is different from 0, at least one bit error has been detected. The frame counter is used by the G-PON-like packet checker to identify the condition when one or more packets have been skipped. This condition is indicated by the line FRAME_ERR being pulsed to 1. The signal FRAME_ALIGN is always NOT(FRAME_ERR) and is thus a redundant signal.

BCDR Simulation Signals Description

Table 3 provides a description of the signals displayed in the simulation window.

Table 3: **BCDR Simulation Signals**

Signal Name	Type	Description
FRAME_ALIGN	Bit	This signal is set to 1 by the G-PON-like pattern checker when an out-of-sequence packet is detected.
PRBES_ERR	Bit	All zeros means that no errors are detected in the payload of each packet.
BURST_EN	Bit	When set to 0, the burst detection capability of the BCDR is disabled. This is a debug condition. This signal is connected directly to the BURST_EN input of the BCDR core.
FR_START	Bit	Signal from the G-PON-like packet receiver. FR_START marks the frame start as seen by the G-PON-like packet receiver.
FR_END	Bit	Signal from the G-PON-like packet receiver. FR_END marks the frame end as seen by the G-PON-like packet receiver.
FR_NM	Bit	Frame number as seen by the G-PON-like packet checker. A skipped packet always produces bit errors and forces FRAME_ALIGN to temporarily go to 0.
FR_LOSS	Bit	This is always NOT (fr_align) and is thus a redundant indication.
DT_IN	Bit	Serial data in.

Table 3: **BCDR Simulation Signals *(Cont'd)***

Signal Name	Type	Description
RIT_INT	Time	Transport delay applied to DT_IN.
/INST_G-PON_PSEUDO_GEN/CNT	16-bit Vector	Unsigned packet number as injected by the G-PON-like packet generator.
/INST_G-PON_PSEUDO_CHECKER/CNT	16-bit Vector	Unsigned packet number as seen by the G-PON-like packet checker.
EN_HAMMER	Bit	When set to 1, consecutive transmitted packets have a 0.5 UI phase difference.
AVE_SEL	2-bit Vector	See Table 1, page 5 for description.
PHE_BST_DRU	16-bit Vector	Signed output. This is the phase profile of the incoming data as seen by the BCDR.
PHE_BURST_AVE	16-bit Vector	Average of PHE_BST_DRU.
DLY_INC	Time	Additional transport delay applied to serial input data.

BCDR Simulation Sequence

The simulation is divided in two parts:

1. Up to about 1.5 ms into the burst, the debug signal BURST_EN is set to 0 (debug condition).
2. Afterwards, it is set to 1 (default).

Figure 7 illustrates the BCDR behavior when the burst injection capability has been disabled. It shows a zoom of the simulation across the BURST_EN change from 0 to 1.

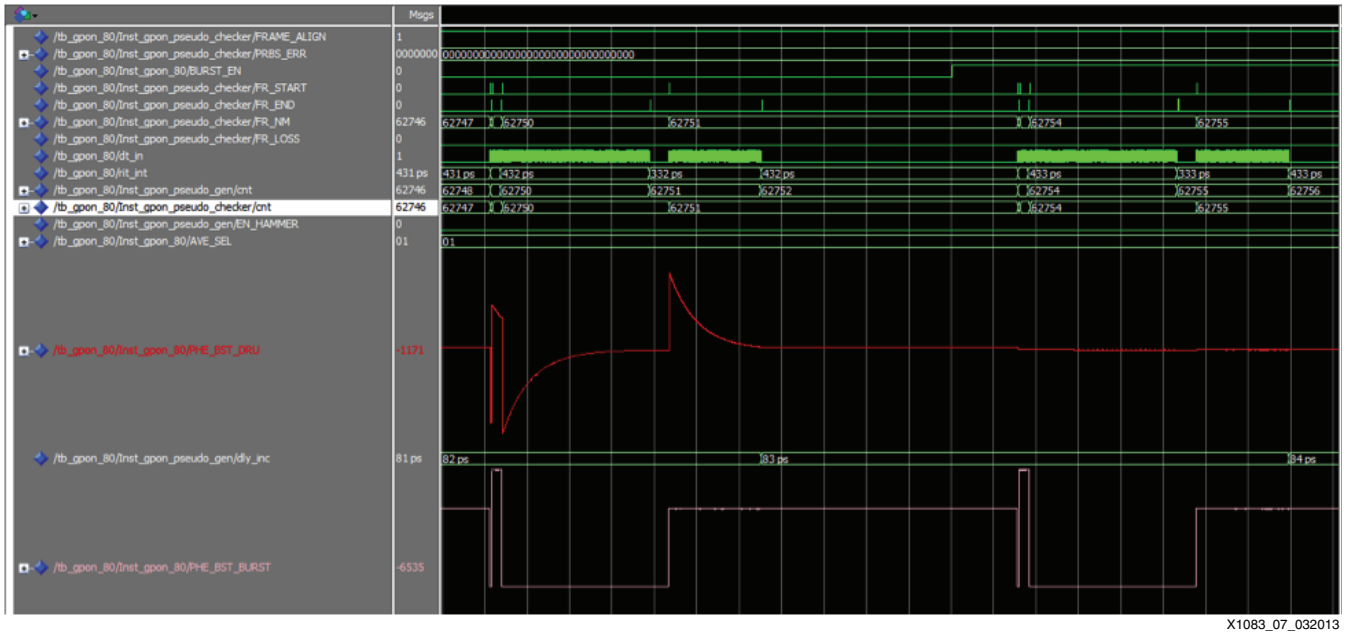

Figure 7: **Effect of the Burst Detection Capability**

In the left half of the simulation, the burst detection capability is not active (debug condition). The phase error appearing in red on the simulation screen is reduced with an exponential transient by the tracking loop in the lower branch. The pink trace is the phase profile of the incoming data as seen by the BCDR. On the right hand side, the preamble detector is active and the NCO is steered at the beginning of each packet to the phase of the incoming data. The

phase error is always kept close to 0 by the tracking loop. This behavior is acceptable for a continuous CDR but not for a BCDR because the bits at the beginning of each packet risk being decoded incorrectly.

The portion of the simulation on the right side shows the effect of BURST_EN = 1 (operating condition). The tracking loop is steered at the beginning of the packet so that the starting phase error is minimized. It is the task of the BCDR to decode correctly all bits in the packet, sacrificing only the bits of the preamble.

Only when BURST_EN is set to 1, two signals from the G-PON-like pattern checker help to decide whether bit errors have been seen and no frames have been lost:

- prbs_err: Must always be 0 because it is set to 1 when at least one bit error has been detected.
- Packet number: This value increases over time. Being the payload of a truncated PRBS, a bit error or simply a packet skip is detected through a PRBS error.

When the counter is increasing over time and prbs_err is 0, the system is working correctly. The condition in which prbs_err is only equal to 0 is not enough to conclude that the system is working, as it might be that no packets are detected. For this reason it is important that the packet number is increasing over time.

In the simulation, four packets of different length are periodically generated every 125 µs. The cyclical nature of the test bench is a simulation choice because the receiver does not use this information.

During simulation, a delay is introduced and increased in 1 ps steps at the beginning of each group of four packets. The intent of increasing the delay is to scan all possible input phases relative to the local clock to verify that the phase detector works unbiased over a 2π radiant. One ps equals 0.9 degrees or 2.5 mUI. The phase profile of the incoming packets moves slowly as the time increases.

While the first packet phase increases by 1 ps at each cycle, the second packet phase is shifted 0.5 UI backwards compared to the first packet. The intent of this test is to stress the receiver with the maximum phase change of 0.5 UI from packet to packet and verify that each frame is still captured with no bit errors. This is the most significant test conducted during the simulation and hardware test. For this reason, it is called the “hammer test” in the context of this application note.

Hardware Test Bench on the KC724 Board

Setting Up the Hardware Test Bench

The directory `implement` contains the script `implement.bat` for WIndows, and `implement.sh` for Linux. The scripts are necessary to implement the BCDR receiver design example based on the Kintex-7 XC7K325T-3FGG900 device hosted on the KC724 characterization board.

The FPGA configuration file `tb_hw.bit` has been generated already and is available for use. After the file is downloaded into the FPGA, the file `tb_hw.cpj` can be used to configure the ChipScope™ Pro analyzer with the virtual input/output (VIO) module and the integrated logic analyzer (ILA) core.

Reference Clock Setup

The programmable clock generator on the KC724 characterization board is used to generate the 155.52 MHz clock in this implementation. These are the steps to program the clock generator using the ChipScope Pro analyzer:

1. Program the FPGA with the bitfile and the configuration file under `/superclock2/`.
2. Open the VIO console windows.
3. Configure the module as shown in Figure 8.

4. Verify with a scope that all clock generator module outputs are set to 155.52 MHz.
5. Connect the generated reference clocks to refclk1 in Quad 115 for the transmitter.
6. Connect the generated reference clocks to refclk1 in Quad 118 for the receiver.

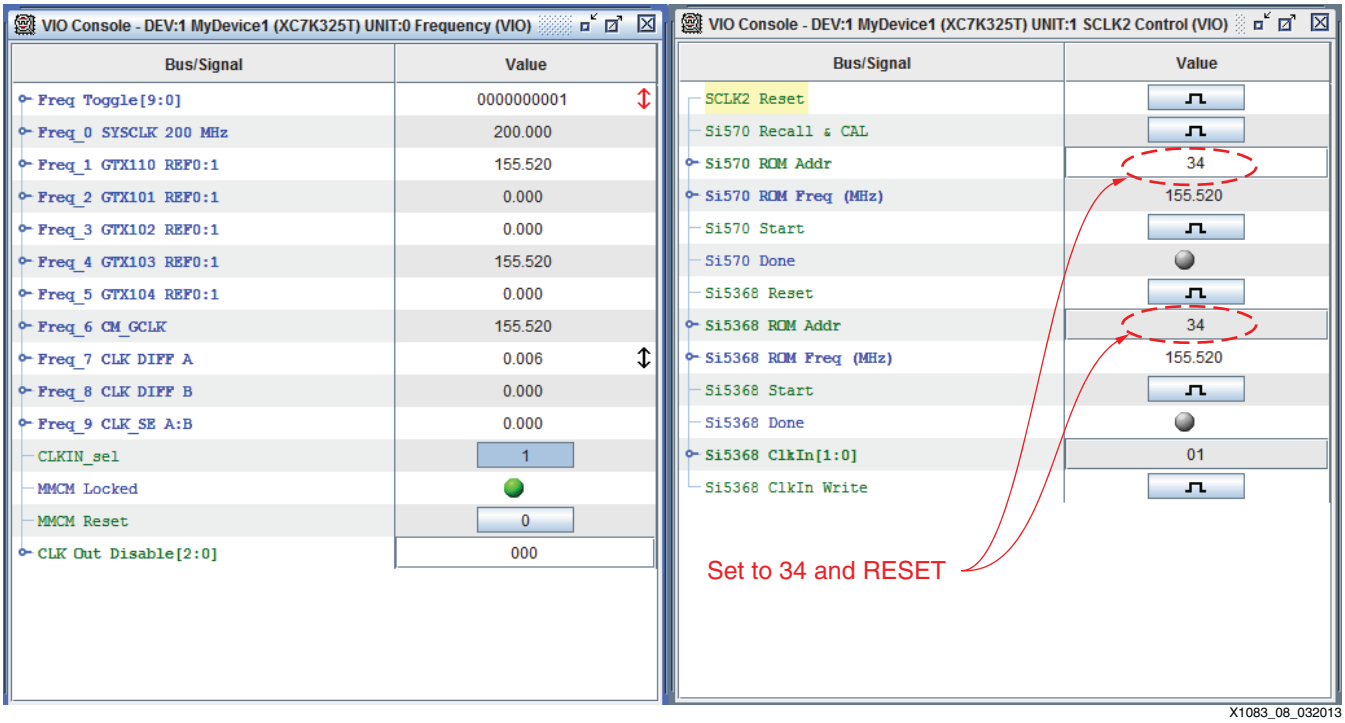

Figure 8: **Programming the Superclock 2 Module**

Note: Both transmitter and receiver must be DC-coupled for a reliable burst packet transfer.

Additional Resources

These probe signals are available to help debug the core:

- RXUSERCLK2 on AH29
- TXUSERCLK2 on AG29
- ASYNC signal on D18

Note: The SYNC signal is a pulse made available each 125 µs to trigger a real-time scope. All this information can be extracted and cross checked with the UCF file of the top level.

Changing characterization board parameters requires changing only the UCF file because no additional external clock is required.

The design example `tb_hw.bit` and the dedicated ChipScope Pro project file `tb_hw.cpj` are now ready to be loaded from the `implement` directory. Figure 9 shows the VIO console for reference.

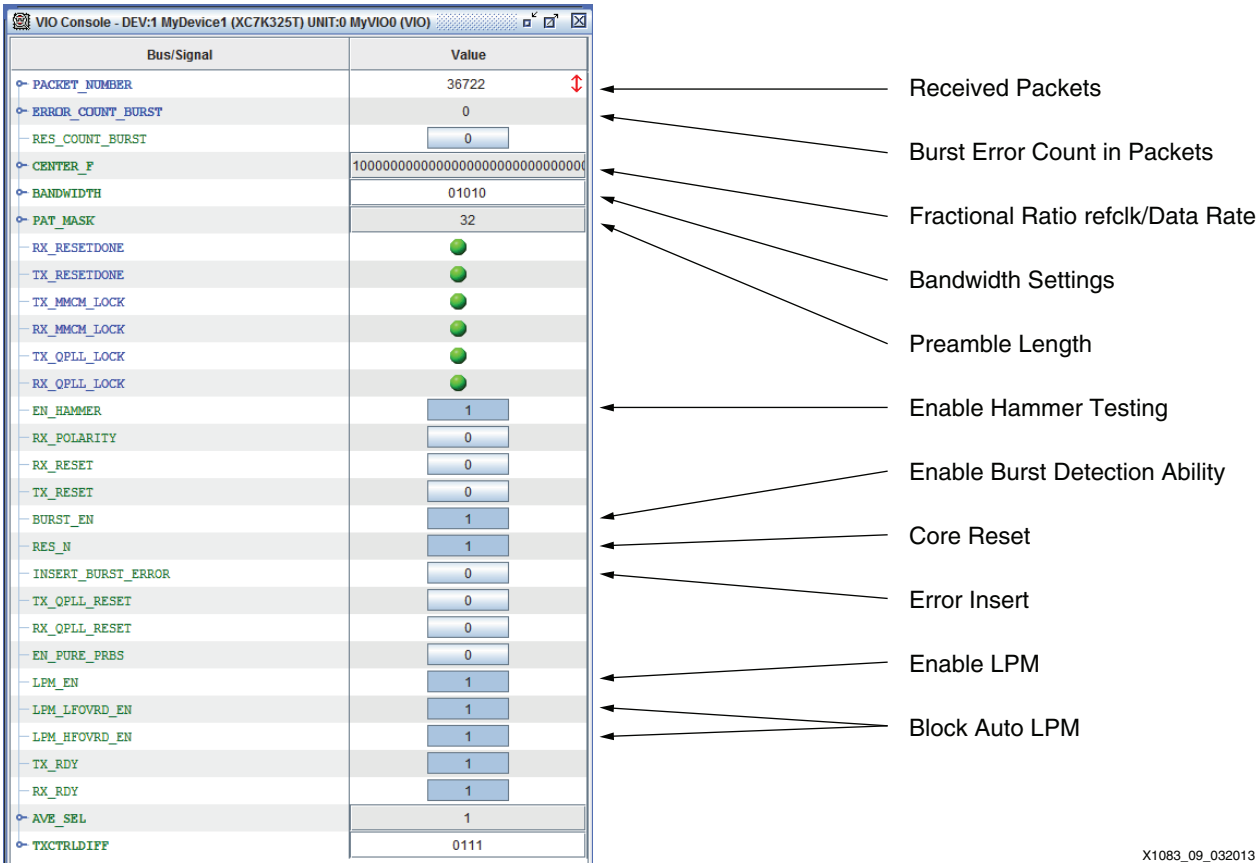

Figure 9: **ChipScope Pro VIO Module for BCDR Test Bench Control**

BCDR Operational Testing

This section describes the hardware test bench structure, its purpose, and the hammer test that it performs.

Hardware Setup

Figure 10 shows the block diagram of the hardware test bench.

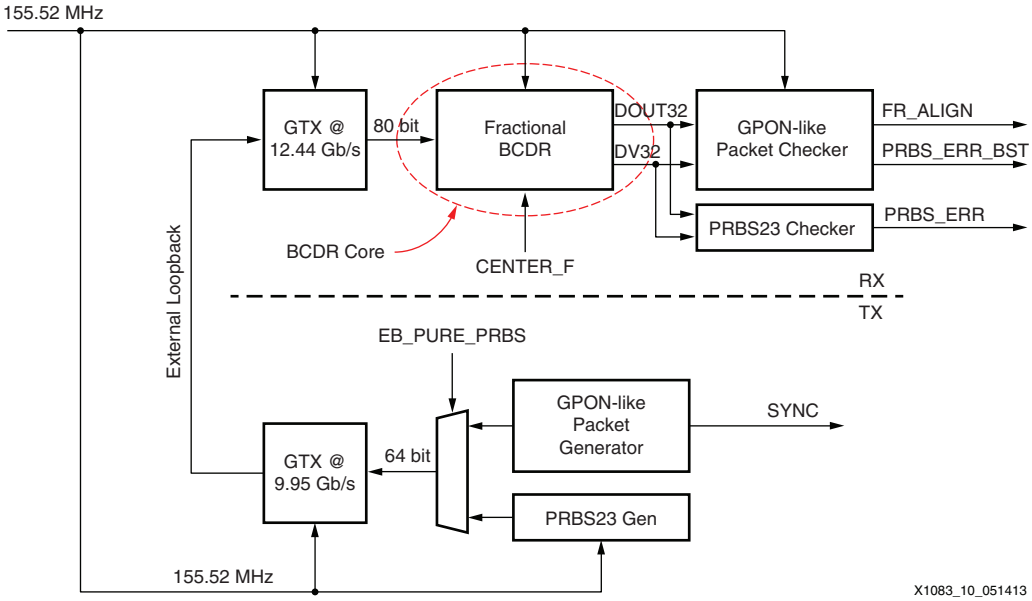

Figure 10: **BCDR Hardware Test Bench Block Diagram**

The hardware setup includes a G-PON-like pattern generator and checker and is able to generate phase jumps between packets of 180 degrees (EN_HAMMER set to 1).

The transmitter can generate either a G-PON-like pattern or a pure PRBS. The receiver can check both. Everything is controlled by a ChipScope Pro module. Errors from the checkers can be used as a trigger for an ILA module already embedded into the design.

Note: For a review of the theoretical and practical aspects of PRBS use cases, see *An Attribute-Programmable PRBS Generator and Checker* (XAPP884).

The G-PON packet generator also generates a synchronism every 125 µs to allow for a real-time scope to be triggered to easily observe each transmitted packet. This is available on pin K39 of the FPGA. In this example design, the BCDR receiver uses SerDes 0 in Quad 118, while the G-PON-like transmitter uses SerDes 0 in Quad 115. Two Quads are needed because the QPLLs operate at different frequencies.

Executing Hammer Test

These steps describe how to execute the BCDR circuit hammer test:

1. Reset transmitter: In sequence, pulse the QPLL reset for the transmitter and pulse the TX RESET. Afterwards, TXRESETDONE should be High. TX RESET is controlled and TXRESETDONE is monitored by the ChipScope Pro analyzer.
2. Reset receiver: In sequence, pulse the QPLL reset for the receiver and pulse the RX RESET. Afterwards, RXRESETDONE should be High. RX RESET is controlled and RXRESETDONE is monitored by the ChipScope Pro analyzer.
3. Transmit a pure PRBS: Pulse RES_N. This resets all the test benches. The GT transceivers are not affected by this reset.
4. Set EN_PURE_PRBS to 1: On a scope, check that a PRBS signal is transmitted at 2.488 Gb/s.
5. Enable LPM equalizer and disable auto adaptation:
    a. Set LPM_EN to 1 in the VIO console. This can also be left permanently set to 1 in the ChipScope Pro analyzer.
    b. Set LPM_LF_OVDR_EN = LPM_HF_OVDR_EN = 1 to ensure that the receiver equalizer is at a fixed level.

c. Monitor the PRBS_CHECK signal in the ILA to check whether the PRBS is received correctly. No errors are expected.

d. Run a PRBS loopback test to ensure that the board is ready to start testing the BCDR circuit.

Note: The BCDR is already working, but the upper branch never detects a preamble in the data stream.

e. Set EN_PURE_PRBS to 0 to switch on the burst pattern.

Note: As described in BCDR Simulation Sequence, page 9, this demonstration works correctly when PACKET_NUMBER increases over time and BURST_ERROR_COUNTER is frozen. To reset BURST_ERROR_COUNTER to 0, pulse RESET_BUSRT_COUNT in the ChipScope Pro analyzer VIO.

6. Enable/disable hammer testing: The transmitter sends bursts of varying lengths. Each packet contains a truncated PRBS, meaning that as soon as a bit error occurs or a packet start is not detected, the burst checker indicates an error. An error counter is provided for long-term testing.

   a. Use EN_HAMMER to either enable or disable the hammer testing. BURST_EN activates the BCDR's capability to track bursts. Set BURST_EN to 1 to get no errors during the hammer test, which is expected.

   b. Track the phase profile of the incoming packets and the estimated phase of the BCDR in the plot window available in the ChipScope Pro analyzer. If the hammer test is not enabled, the BCDR performs with no errors, independent of the burst acquisition capability being on or off, as expected.

Analyzing BCDR Test Results

This section presents screen captures of displayed measurement results to enable the reader to understand the BCDR operation and interpret test results.

Note: This section refers to screen artifacts as defined by their color, such as red or blue traces or lines. For clarity, the reader should either view this content on a color display or obtain a color printout of this document.

BURST_EN Set to 0

Figure 11 shows the measured packet phase and BCDR phase error when BURST_EN is set to `0` and the bandwidth is set to `1010`. The blue line is the phase profile of the incoming packets as measured by the BCDR. On the vertical axis, 1,024 steps correspond to 360 degrees. The resolution of the 5X oversampler is limited to 72 degrees.

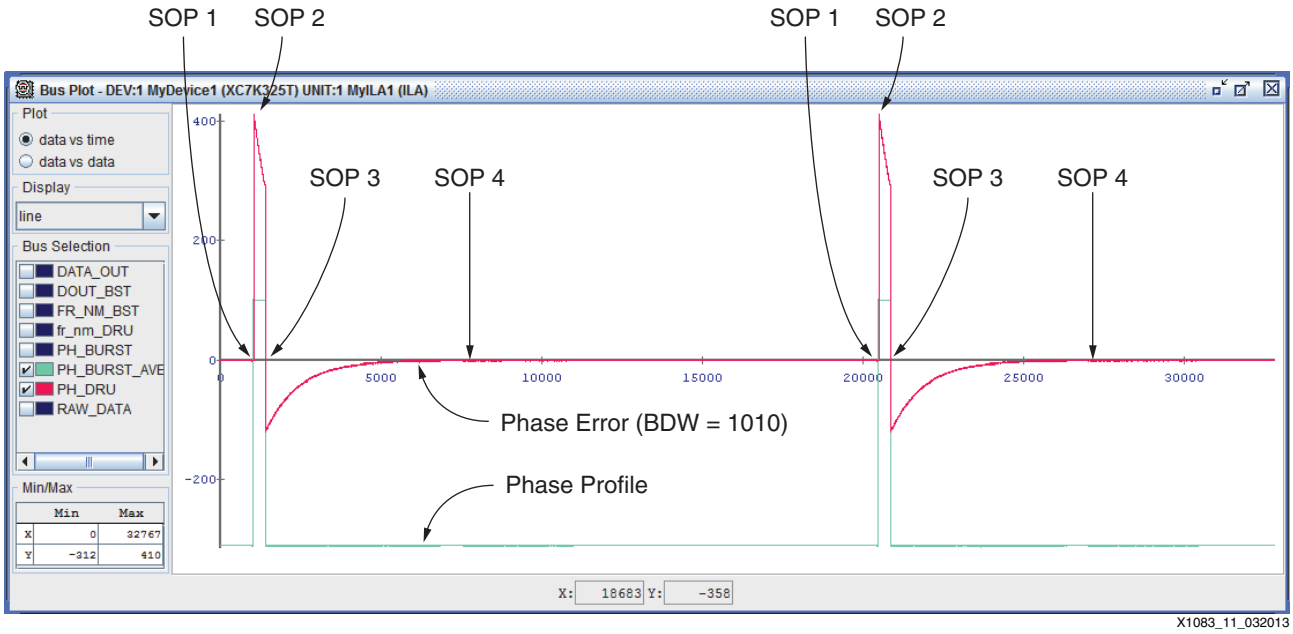

Figure 11: **Packet Phase (Blue) and BCDR Phase Error (Red)—BURST_EN Set to 0, Bandwidth Set to 1010**

The measurements are performed on four packets. For each packet, the start of packet (SOP) is shown. There is a 180 degree phase jump between packets 1 and 2 and between packets 2 and 3. This is visible in the phase profile. When EN_HAMMER is set to 0, the phase jumps are removed and the blue line is expected to become flat.

BURST_EN is set to 0, thus the BCDR detects bursts (PL_O is set to 1 at the beginning of each packet) but does not preset the NCO. Thus, the phase error in red is large at the packet start and is slowly corrected by the BCDR over time. The exponential process is visible on the red line, which is the phase error on the output of the phase detector of the lower branch. The exponential time constant is digitally controllable by the BDW port.

For example, Figure 12 shows the same measurement as Figure 11 but with the bandwidth set to `1011`, and thus with the CDR bandwidth reduced by half.

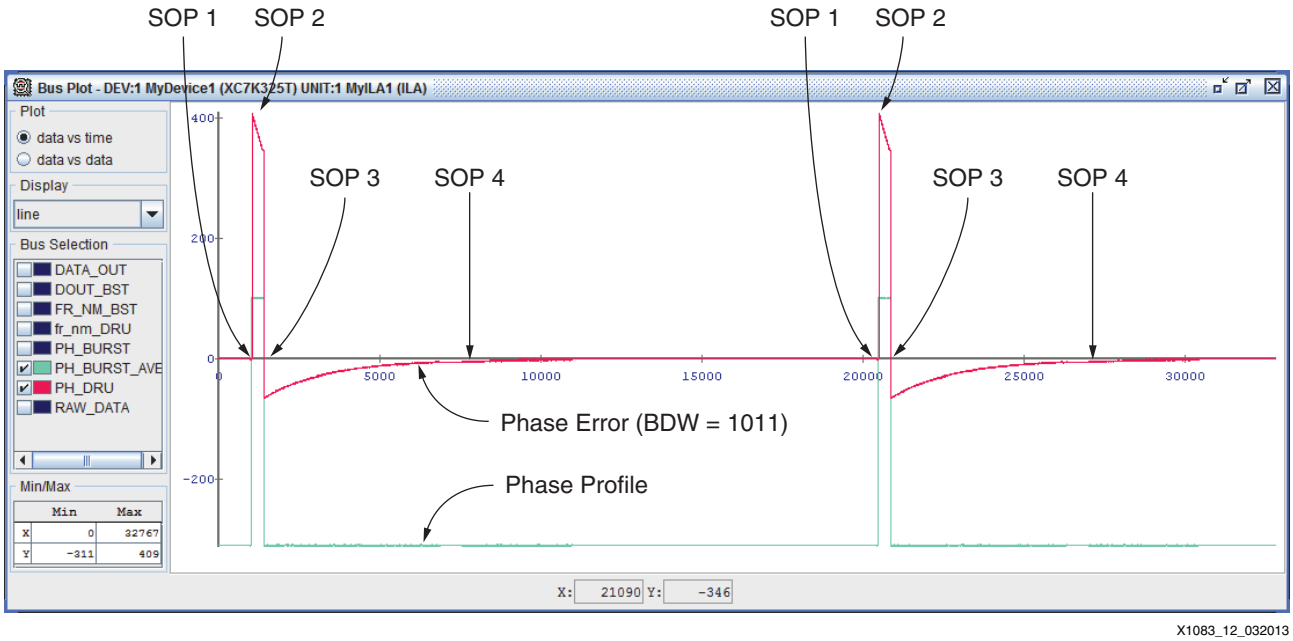

Figure 12: **Packet Phase (Blue) and BCDR Phase Error (Red)—BURST_EN Set to 0, Bandwidth Set to 1011**

BURST_EN Set to 1

Figure 13 shows how in the presence of the hammer test, the BCDR does not see any phase error. This is because the internal NCO is steered in one single clock cycle to the estimated start of packet phase as soon as the preamble is detected. Equivalently, consecutive packets have no phase jump for the lower branch in the BCDR.

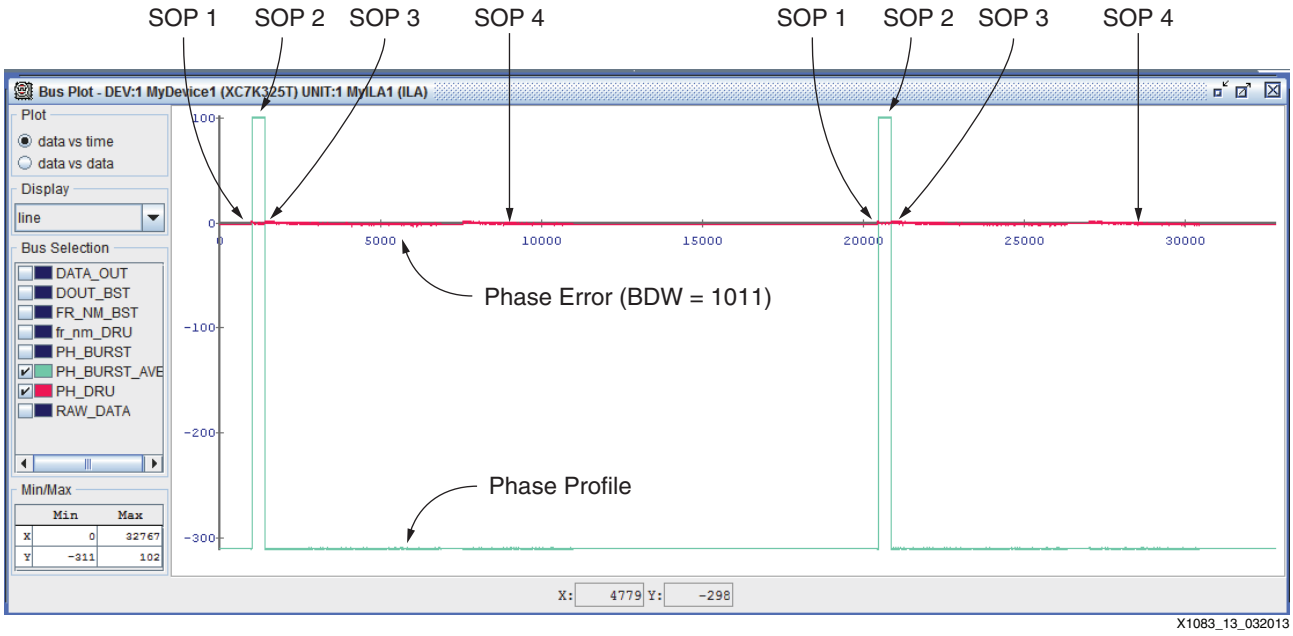

Figure 13: **Packet Phase (Blue) and BCDR Phase Error (Red)—BURST_EN and HAMMER_TEST Set to 1**

In the cases illustrated in Figure 11 and Figure 12, the system is prone to errors. In the case of Figure 13, however, the system is error-free.

Figure 14 is a zoom of Figure 13 showing only three packets. The blue and red measurement lines can be noisier than shown in the figure. This sampling noise depends on the relative phase between the free running oversampler and the data edges, and its presence is expected. The sampling noise is not a symptom of undesirable operation as long as it is less than the oversampling resolution—in the case of 5X, the resolution is 0.2 UI.

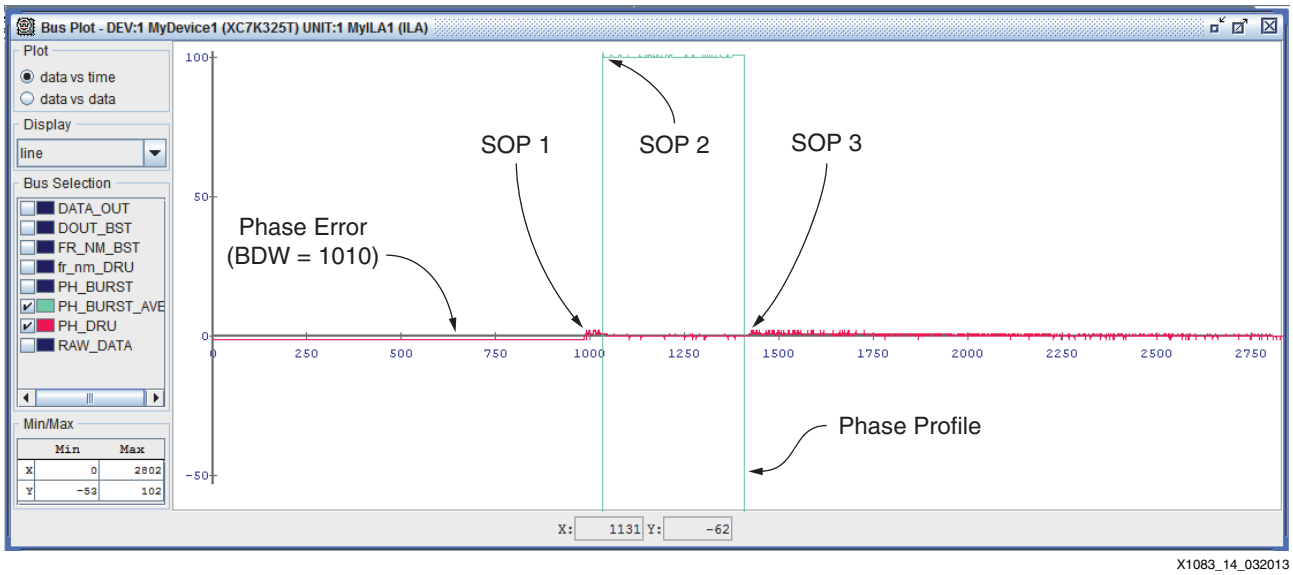

Figure 14: **Zoom of Figure 13, SOP 4 Not Visible**

Reference Design

The reference design files for this application note can be downloaded from:

https://secure.xilinx.com/webreg/clickthrough.do?cid=343154

Table 4 shows the reference design checklist.

Table 4: **Reference Design Checklist**

Parameter	Description
General	
Developer name	Paolo Novellini, Massimo Chirico
Target devices (stepping level, ES, production, speed grades)	Kintex-7 and Virtex-7 devices
Source code provided	Yes (NGC partial)
Source code format	VHDL
Design uses code/IP from existing Xilinx application note/reference designs, CORE Generator™ software, or third-party	Yes
Simulation	
Functional simulation performed	Yes
Timing simulation performed	No
Test bench used for functional and timing simulations	Yes
Test bench format	VHDL
Simulator software/version	ModelSim SE 10.0C or higher
SPICE/IBIS simulations	No

Table 4: **Reference Design Checklist *(Cont'd)***

Parameter	Description
Implementation	
Synthesis software tools/version	XST 14.4
Implementation software tools/versions used	ISE® Design Suite 14.4 or higher
Static timing analysis performed?	Yes
Hardware Verification	
Hardware verified?	Yes
Hardware platform used for verification	KC724 characterization board

Table 5 provides device utilization data.

Table 5: **Device Utilization Data**

Parameter	Specification/Value	
Device utilization without test bench	Slice register	10,990
	Slice LUT	13,418
	DSP48	11
	RAMB36E1	0
	BUFG	0
	MMCM	0
	GTXE2_CHANNEL	0
Device utilization with test bench	Slices register	14,028
	Slice LUT	16,339
	DSP48	11
	RAMB36E1	257
	BUFG	9
	MMCM	2
	GTXE2_CHANNEL	2
Characterization board for test bench	KC724 characterization board, rev. A or later	
Target silicon	Kintex-7 devices, -3 speed grade	
	Virtex-7 devices, -2 or -3 speed grade	

Conclusion

This implementation of a BCDR is well suited to the Kintex-7 and Virtex-7 FPGAs due to the low required minimum oversampling ratio and the highly pipelined oversampling technology. The speed grade requirement is driven only by the GTX transceivers, which must run at 12.44 Gb/s.

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
06/14/2013	1.0	Initial Xilinx release.
06/17/2013	1.0.1	Updated URL to download reference design.
01/17/2014	1.0.2	Corrected typographical errors in Executing Hammer Test.

Notice of Disclaimer

Automotive Applications Disclaimer