Ain Shams University,
Faculty of Engineering,
Computer and Systems Engineering

Digital Design and Verification For SerDes System

Team members

Hassan Khaled Hassan

Abdelmagid Mohamed Abdelmagid

Alaa Salah Abd-Elfattah

Under Supervision of:

Dr. Ahmed M. Zaki

Associate professor,Ain-shams University, Egypt

Sponsored By:

(https://ic-pedia.com)

Contents

Table of Figures

GitHub Repo

https://github.com/HassanKhaled11/SerDes_GP-ICpedia

ABSTRACT

This document is the graduation project report prepared by senior year's students in Computer and System department of Faculty of Engineering Ain Shams to discuss the USB (universal serial bus) Superspeed also known as USB 3.0, it introduces significantly higher data transfer rates and enhanced capabilities compared to its predecessors. The USB SuperSpeed standard employs advanced serializer and deserializer (SerDes) technologies to enable faster communication between USB devices and hosts.

Serializer and deserializer components play a crucial role in USB SuperSpeed by converting parallel data into serial data during transmission and vice versa during reception. This enables efficient utilization of the available bandwidth and facilitates high-speed data transfer rates.

This document also features Functional verification for USB chip to chip layer using SV(system verilog) and UVM (Universal Verification Methodology) with its phases, hierarchy and components to verify the modules and the functionalities implemented.

Introduction

A Serializer/Deserializer (SerDes) is a pair of functional blocks commonly used in high-speed communications to compensate for limited input/output. These blocks convert data between serial data and parallel interfaces in each direction. The term "SerDes" generically refers to interfaces used in various technologies and applications. The primary use of SerDes is to provide data transmission over a single line or a differential pair to minimize the number of I/O pins and interconnects. The basic SerDes function is made up of two functional blocks: the Parallel In Serial Out (PISO) block (aka Parallel-to-Serial converter) and the Serial In Parallel Out (SIPO) block (aka Serial-to-Parallel converter). There are 4 different SerDes architectures: (1) Parallel clock SerDes, (2) Embedded clock SerDes, (3) 8b/10b SerDes, (4) Bit interleaved SerDes.

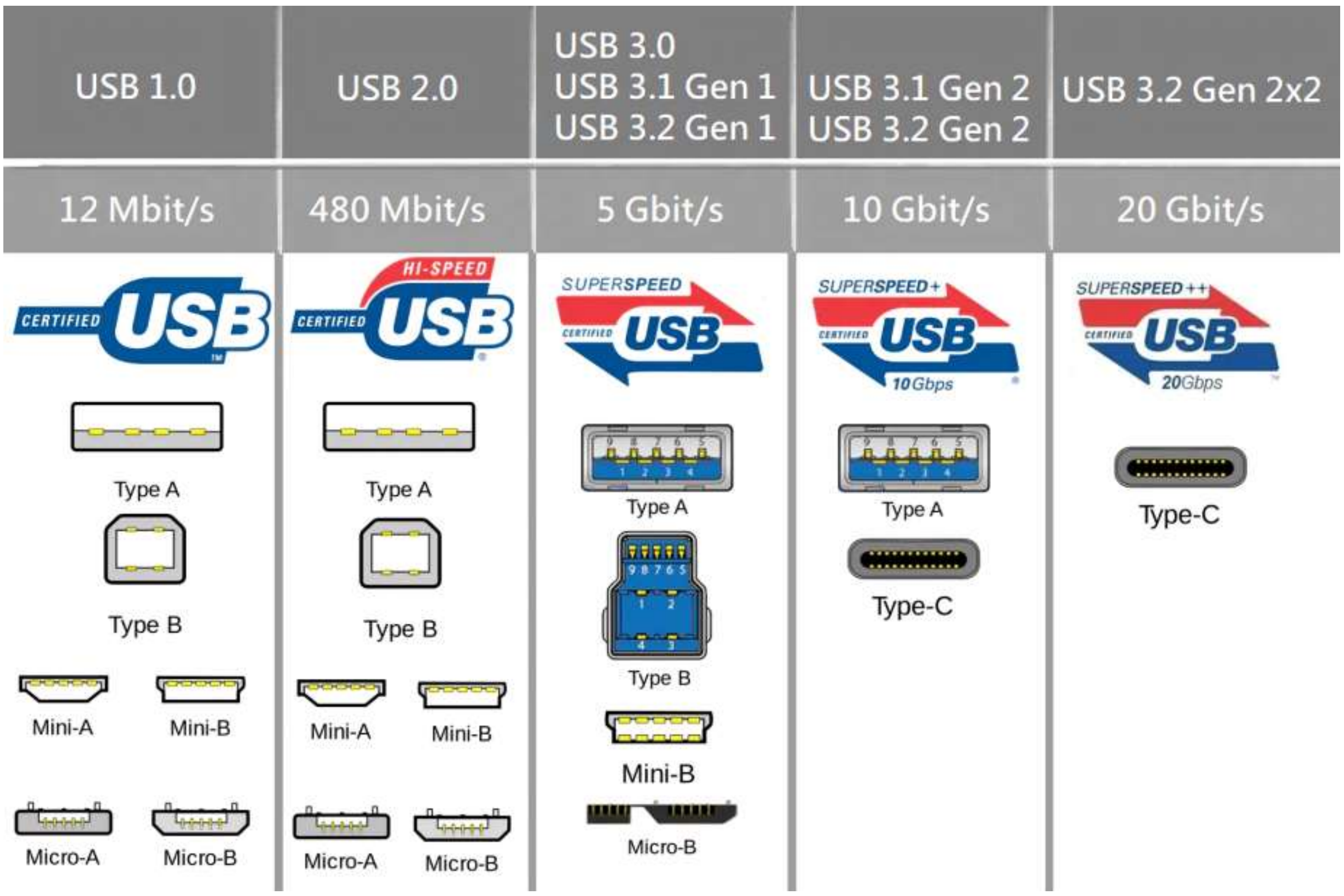
Universal Serial Bus (USB)

The PHY interface for USB SuperSpeed Architecture has passed with many versions till reaching the current version

The document traces the revision history of a USB specification, starting with an initial draft (0.1, 7/31/02) and progressing through multiple versions. It undergoes industry review (0.5, 8/16/02) and provides operational details (0.6, 10/4/02), including timing diagrams (0.7, 11/4/02). Changes are made to the receiver detection sequence (0.8, 11/22/02). The document becomes stable for implementation (0.9, 12/16/02).

Updates reflect the 1.0a Base Spec (0.95, 4/25/03) and include multilane suggestions. The specification stabilizes for implementation (1.00, 6/19/03). Version 1.70 (11/6/05) introduces Gen. 2 PIPE, followed by fixes based on feedback (1.81, 12/4/05; 1.86, 2/27/06). Updates include handling CLKREQ# (1.87, 9/28/06). Minor editorial updates follow (1.90, 3/24/07). Version 2.00 (7/21/07) represents a stable revision.

Version 2.7 (12/31/07) introduces updates supporting USB specification revision 3.0. Subsequent versions (2.71, 2/21/08; 2.75, 2/8/08; 2.90, 8/11/08) include handling SKP and USB SuperSpeed PHY power management. The specification reaches stability for implementation (2.90, 8/11/08). The final version, 3.0 (3/11/09), is declared as the conclusive update. The revisions encompass technical enhancements, compliance with USB standards, and adjustments for evolving USB SuperSpeed modes.



USB PHY with PIPE Schematic

The PHY layer for SuperSpeed is mainly partitioned into PCS(Physical Coding Sublayer) and PMA(Physical Media Attachment).

Common Block which contains PLL that uses REF_CLK to generate Bit_Rate_CLK inaddition to all clks used in the PHY

The PCS have the 8b/10b code/decode and the elastic buffer and RX detection and has more than one clock domain

The PMA have the serial to parallel block,serializer and CDR

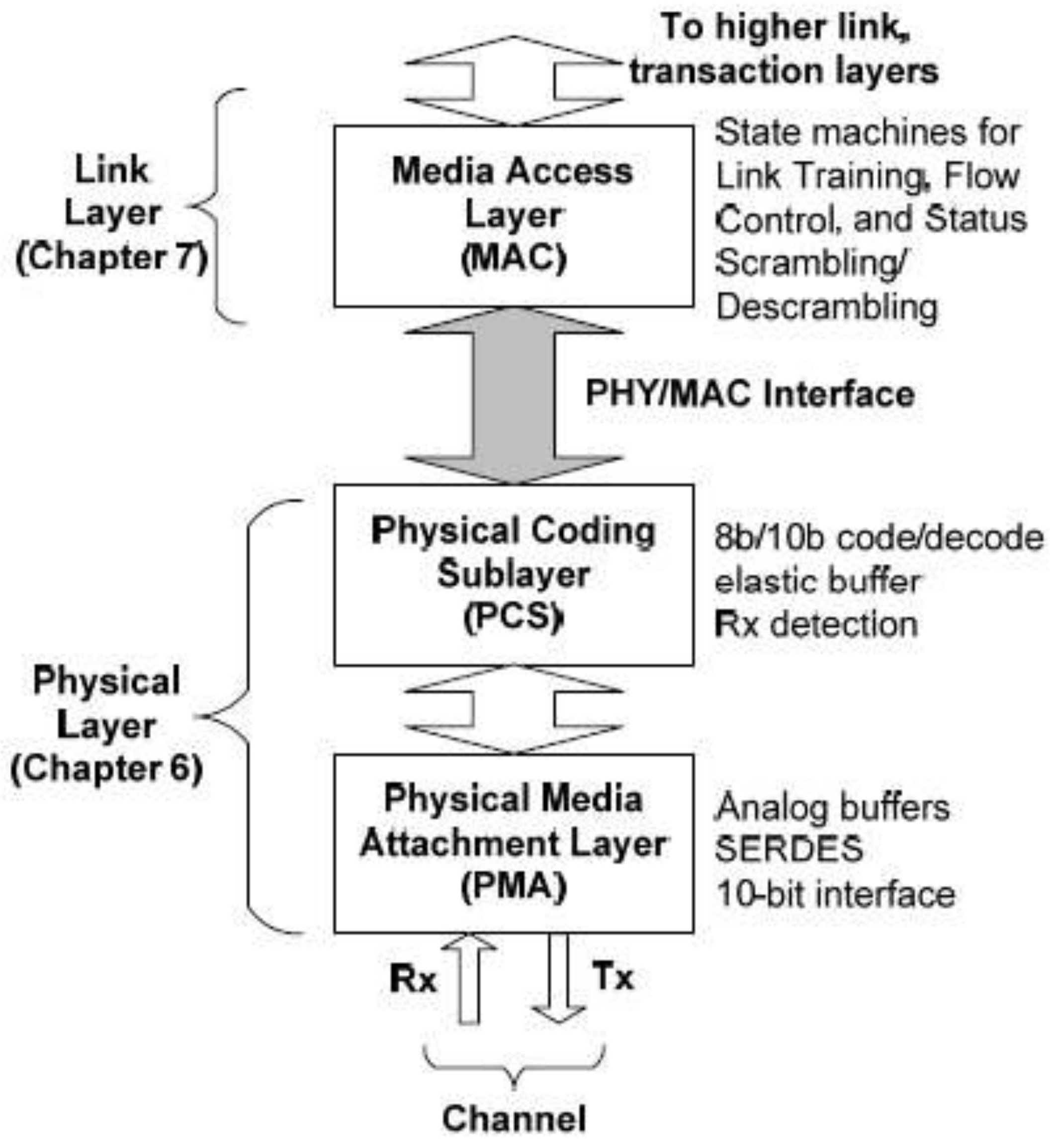

To higher link, transaction layers
Link Layer (Chapter 7)
Media Access Layer (MAC)
State machines for Link Training, Flow Control, and Status Scrambling/ Descrambling
PHY/MAC Interface
Physical Coding Sublayer (PCS)
8b/10b code/decode elastic buffer Rx detection
Physical Layer (Chapter 6)
Physical Media Attachment Layer (PMA)
Analog buffers SERDES 10-bit interface
Rx
Tx
Channel

Figure 1 Superspeed partitions

Physical Interface PCIE and USB(PIPE)

The PHY Interface for the PCI Express and USB SuperSpeed Architectures (PIPE) is intended to enable the development of functionally equivalent PCI Express and USB SuperSpeed PHY's. Such PHY's can be delivered as discrete IC's or as macrocells for inclusion in ASIC designs. The specification defines a set of PHY functions which must be incorporated in a PIPE compliant PHY, and it defines a standard interface between such a PHY and a Media Access Layer (MAC) & Link Layer ASIC. It is not the intent of this specification to define the internal architecture or design of a compliant PHY chip or macrocell. The PIPE specification is defined to allow various approaches to be used. Where possible the PIPE specification references the PCI Express base specification or USB 3.0 Specification rather than repeating its content. In case of conflicts, the PCI-Express Base Specification and USB 3.0 Specification shall supersede the PIPE spec. This spec provides some information about how the MAC could use the PIPE interface for various LTSSM states and Link states. This information should be viewed as 'guidelines for' or as 'one way to implement' base specification requirements. MAC implementations are free to do things in other ways as long as they meet the corresponding specification requirements.

One of the intents of the PIPE specification is to accelerate PCI Express endpoint and USB SuperSpeed device development. This document defines an interface to which ASIC and endpoint device vendors can develop. Peripheral and IP vendors will be able to develop and validate their designs, insulated from the high-speed and analog circuitry issues associated with the PCI Express or USB SuperSpeed PHY interfaces, thus minimizing the time and risk of their development cycles.

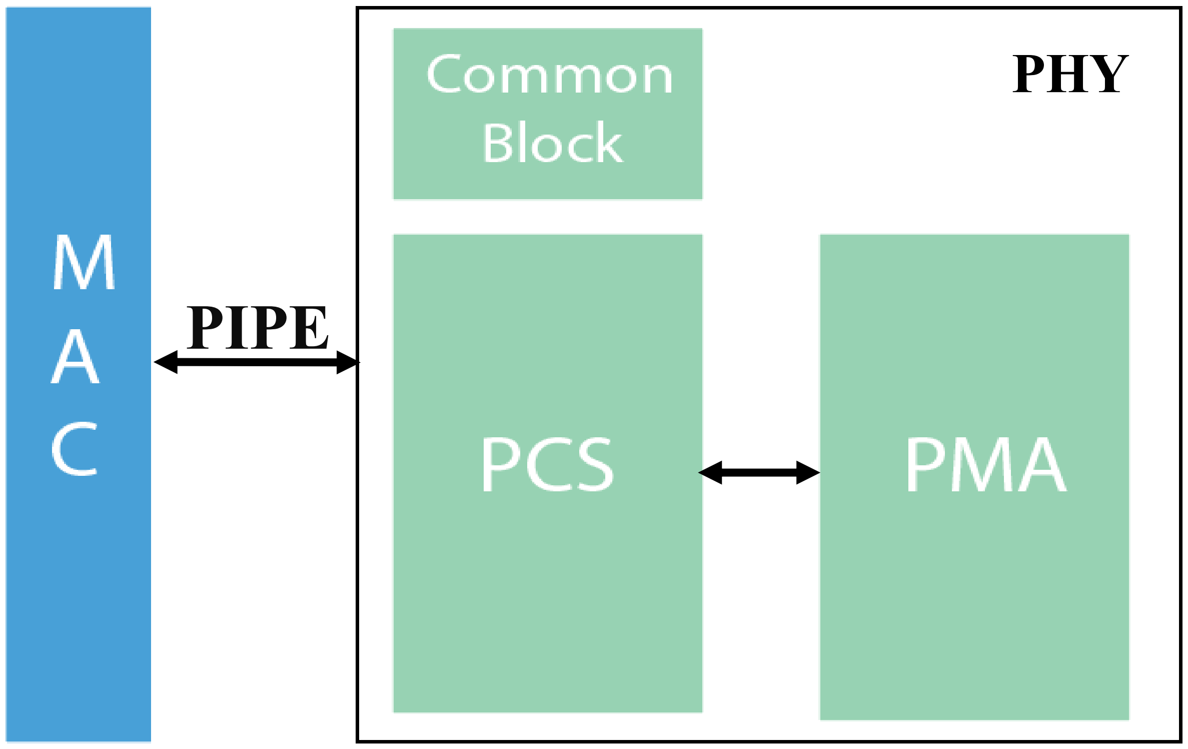


Figure 2 PIPE interface between PHY and MAC

Physical Layer (PHY)

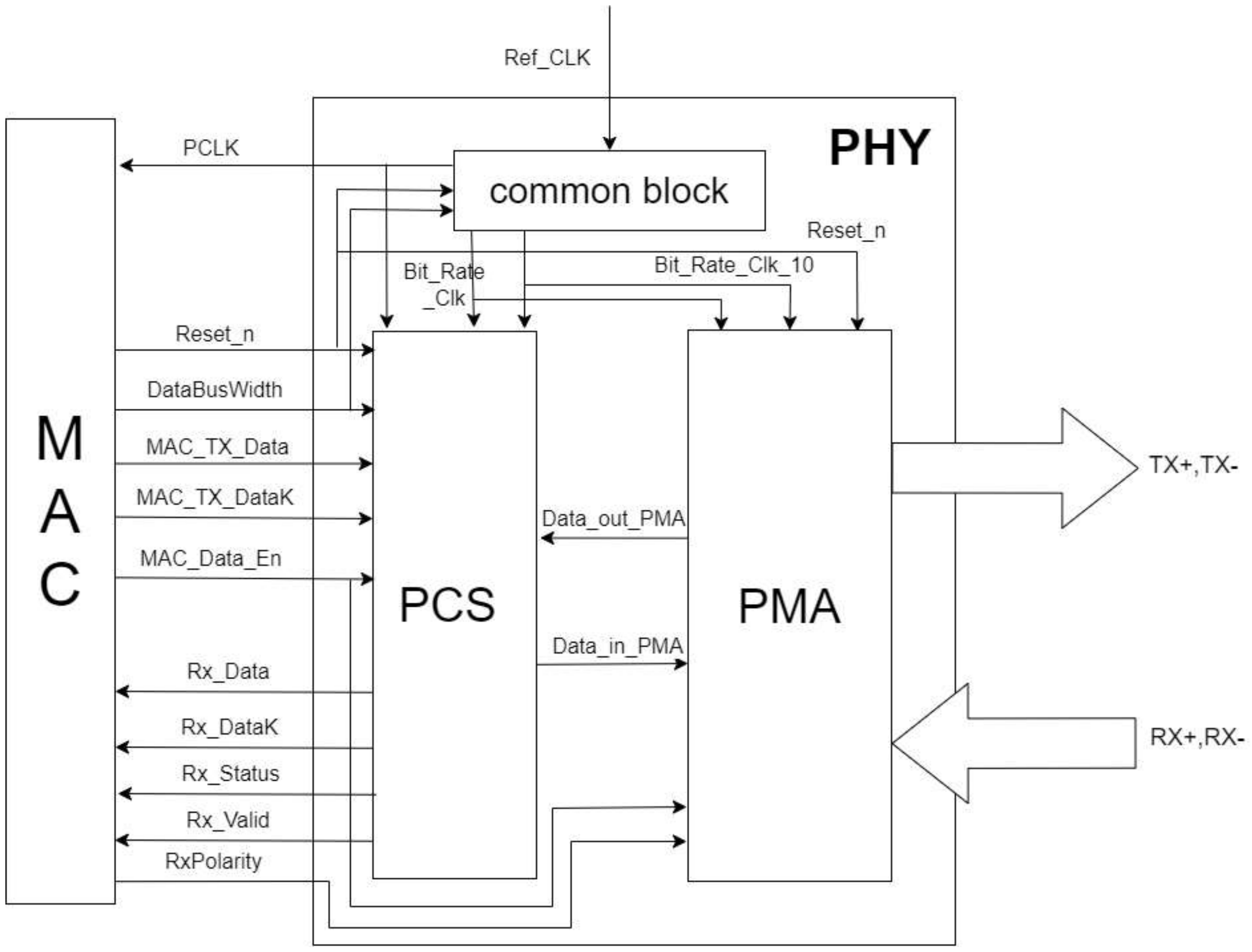


Figure 3 Top view Schematic

PHY Synthesis

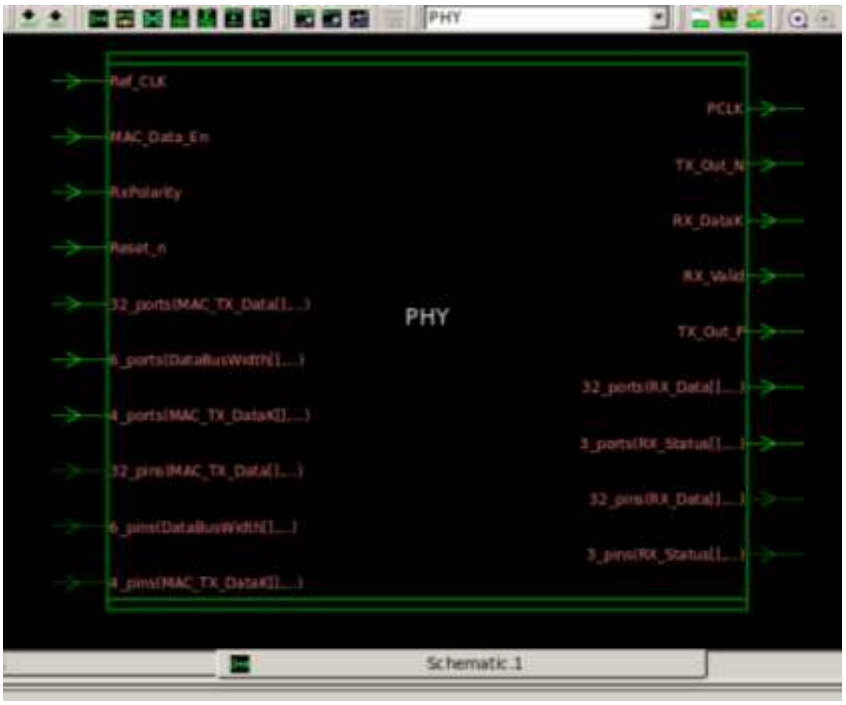
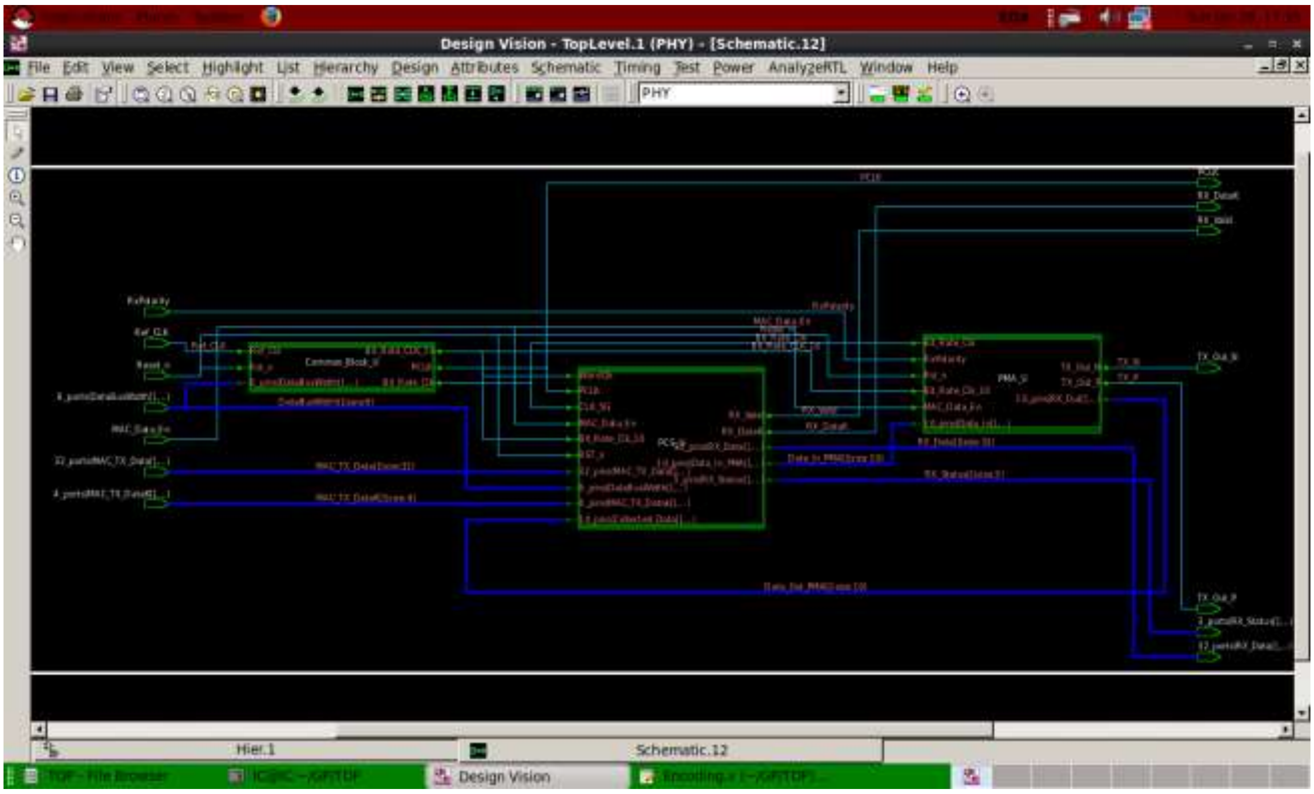


Figure 4 PHY top



Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
PCLK
RX_DataK
RX_Valid
RxPolarity
Ref_CLK
Reset_n
MAC_Data_En
Common_Block_V
PCS_V
PMA_V
TX_Out_N
TX_Out_P
Hier.1
Schematic.12
Design Vision

Figure 5 PHY TX, PHY RX

Physical Coding SubLayer (PCS) Synthesis

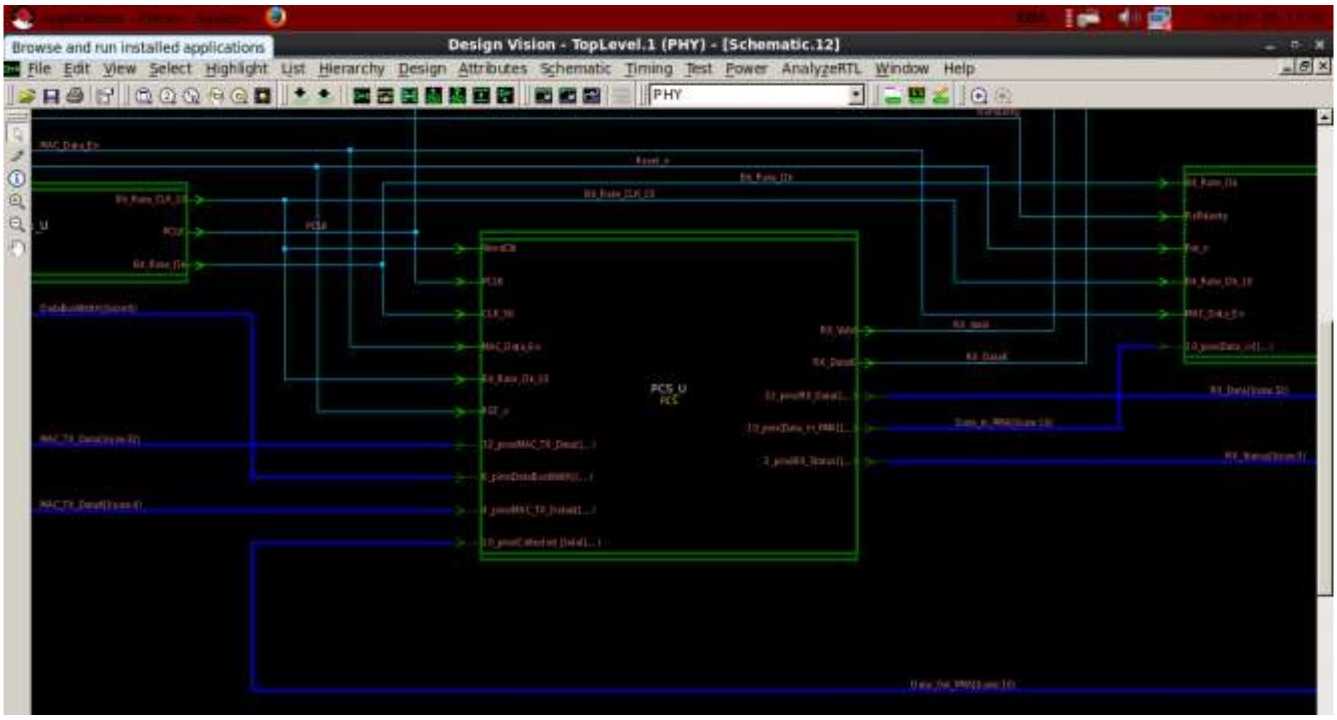

Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
PCS_U

Figure 6 PCS Top

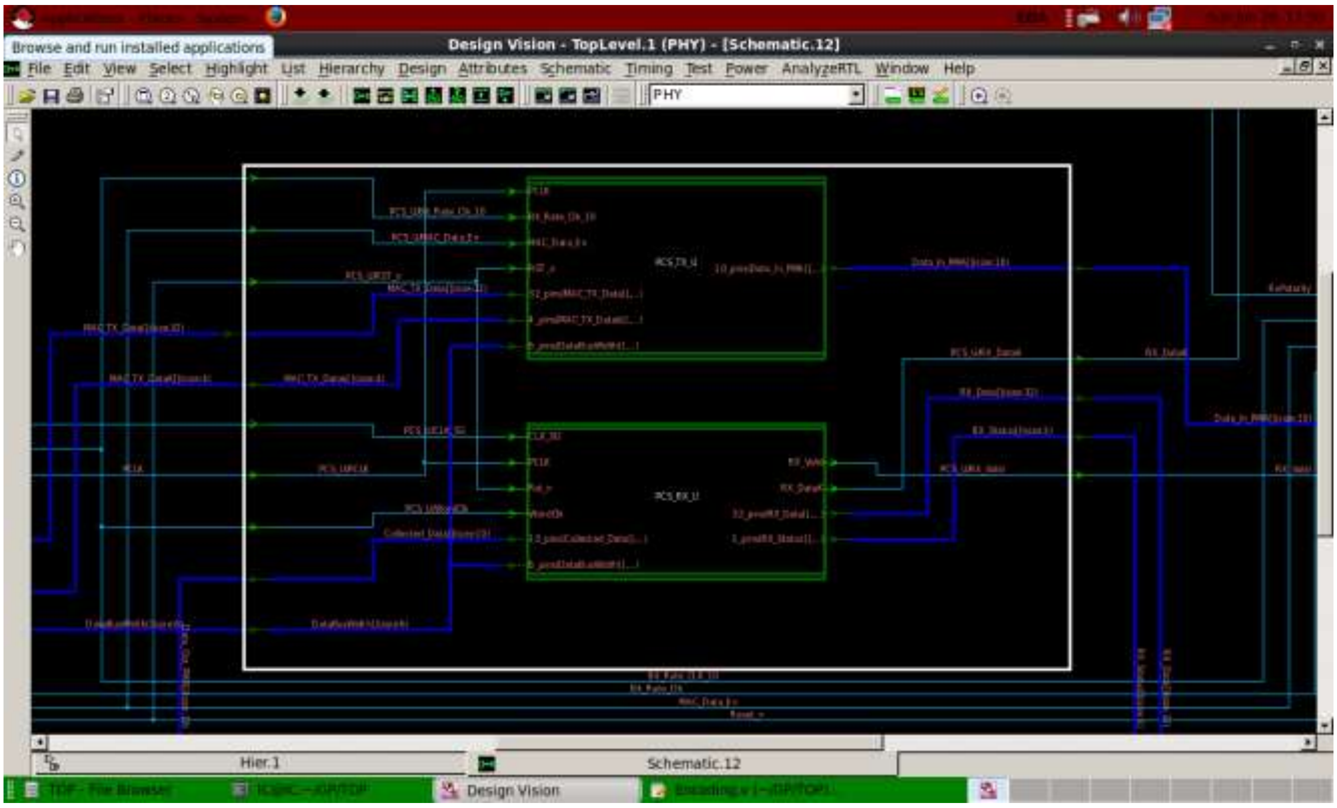

Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
PCS_TX_U
PCS_RX_U
Hier.1
Schematic.12
Design Vision

Figure 7 PCS TX, PCS RX

PCS TX:

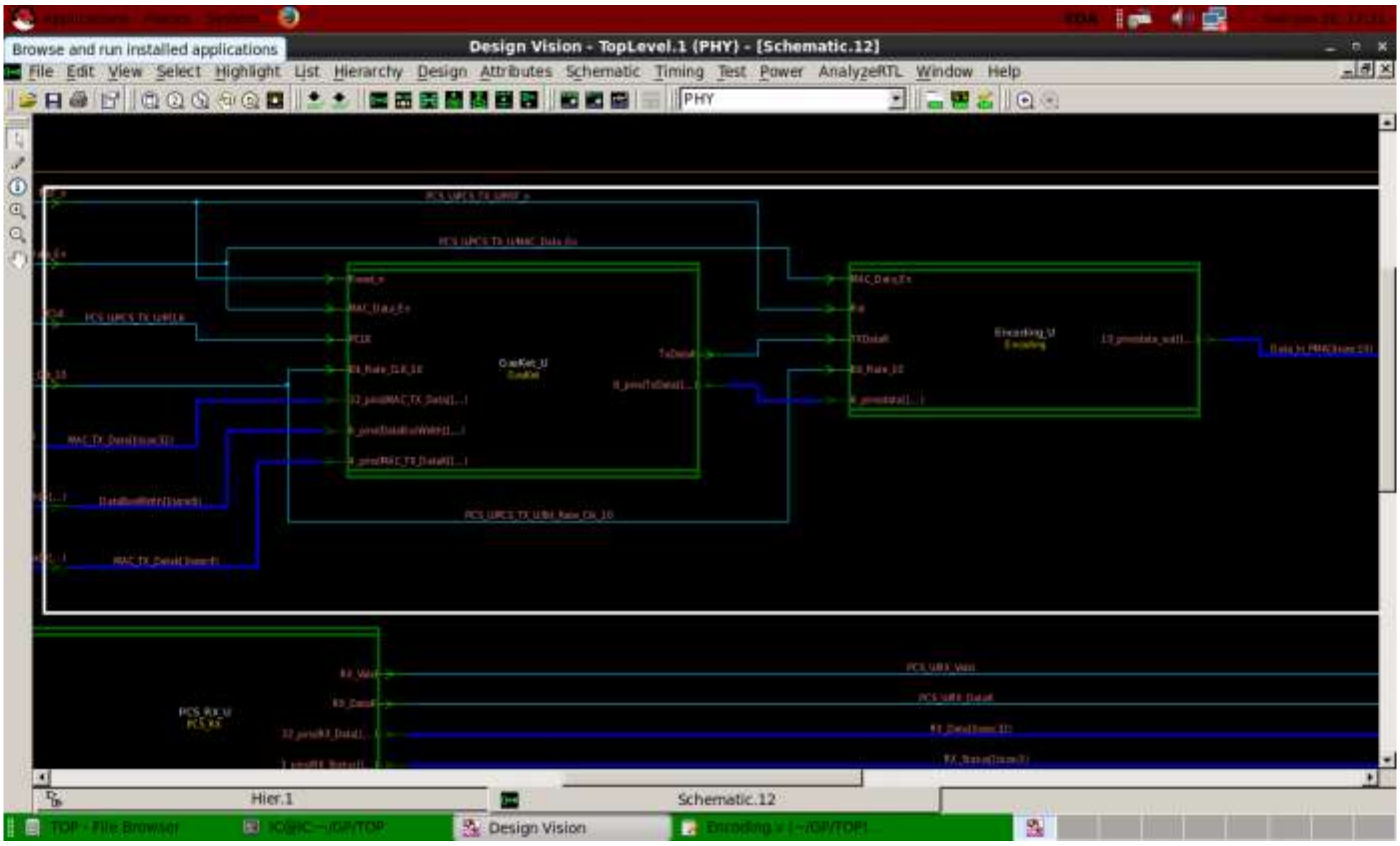

Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
GasKet_U
Encoding_U
PCS_RX_U
Hier.1
Schematic.12
Design Vision

Figure 8 PCS TX

PCS RX

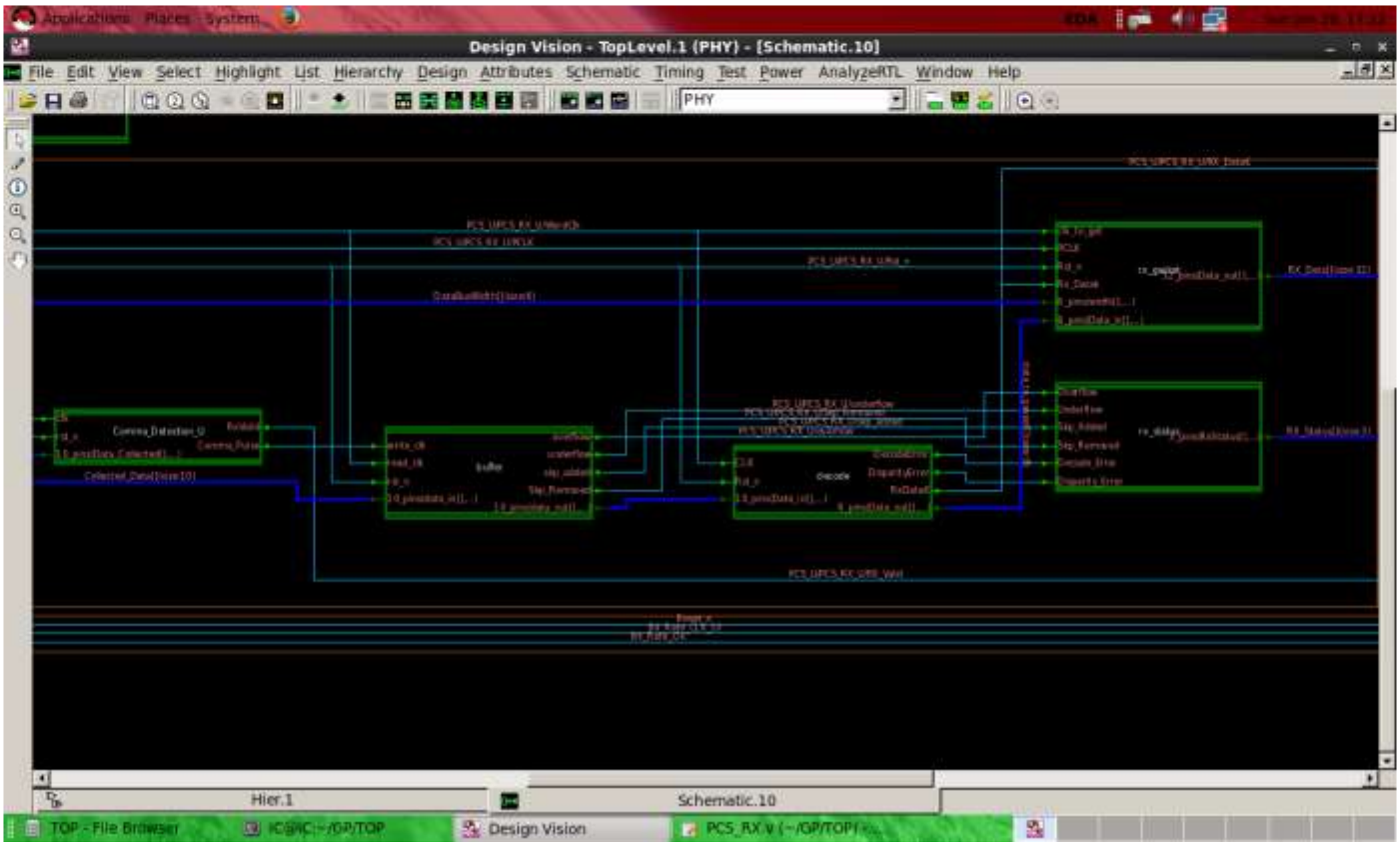

Applications Places System
Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
buffer
decode
Hier.1
Schematic.10
TOP - File Browser
IC@IC:~/GP/TOP
Design Vision
PCS_RX.v (~/GP/TOP/)

Figure 9 PCS RX

Physical Media Attachment (PMA) Synthesis

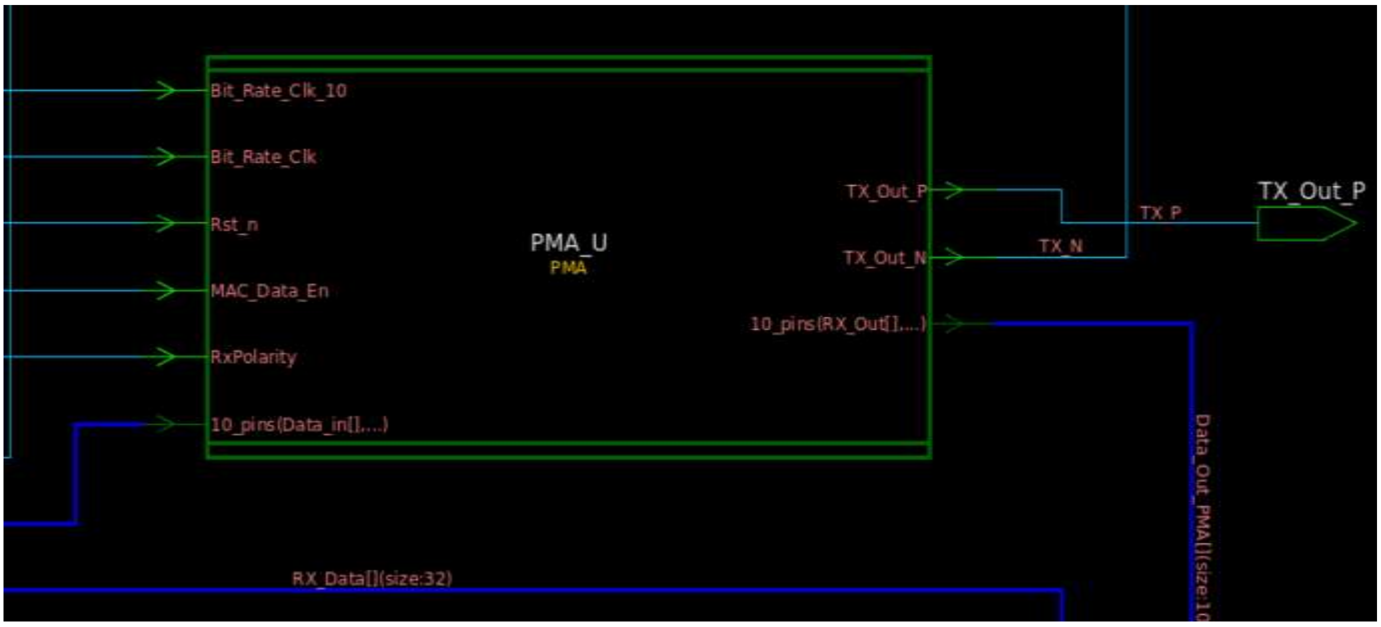


Figure 10 PMA Top

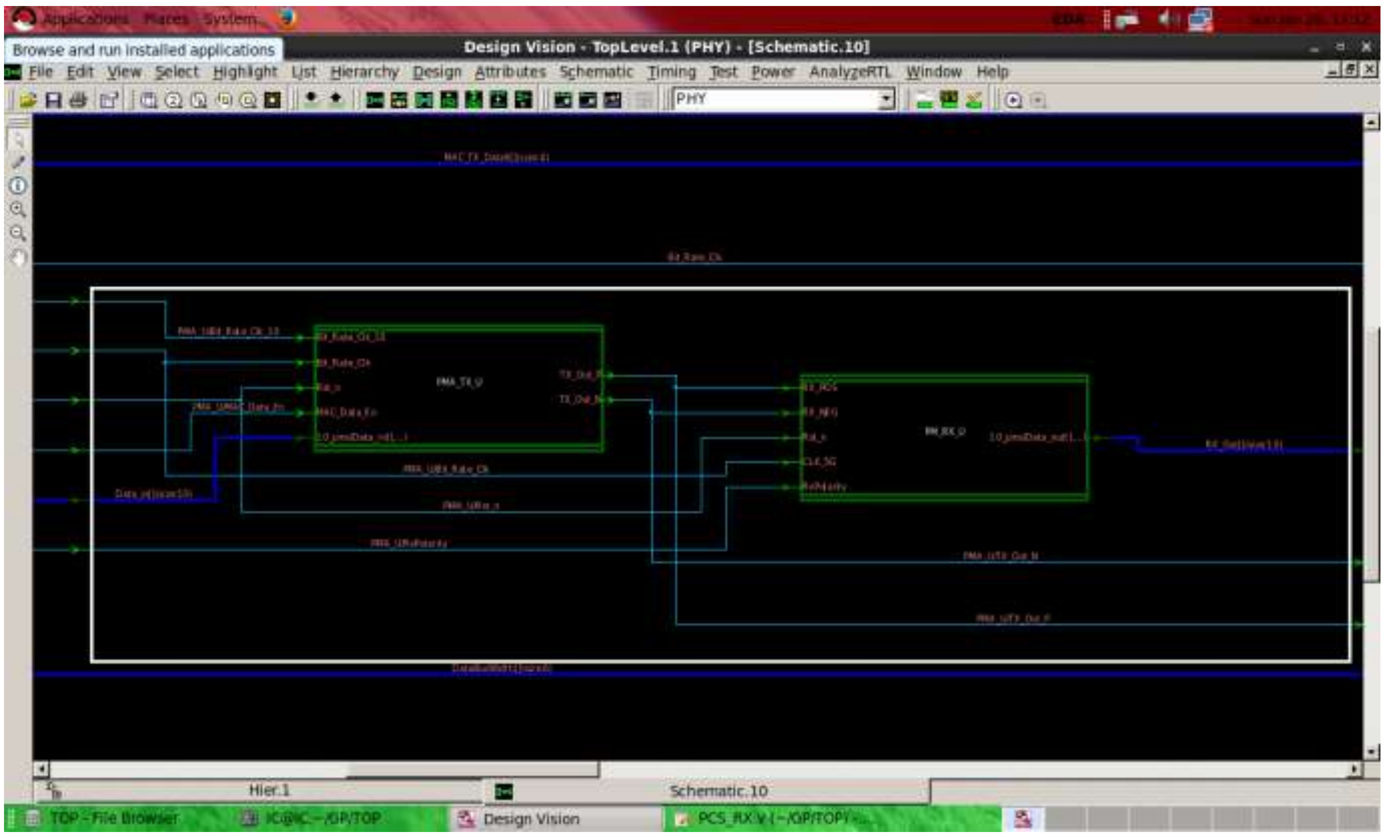

Applications Places System
Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
PMA_TX_U
PMA_RX_U
Hier.1
Schematic.10
TOP - File Browser
Design Vision

Figure 11 PMA TX, PMA RX

PMA TX

Name	Direction	Width	Description
Bit_Rate_CLK	Input	1	Serial CLK(5G)
Bit_Rate_CLK_10	Input	8	Word CLK
Rst_n	Input	1	Active Low Reset
Data_in	Input	10	Encoded Data
MAC_Data_En	Input	1	Enable PMA
TX_Out_P	Output	1	Serial bit sent in positive lane
TX_Out_N	Output	10	Serial bit sent in negative lane

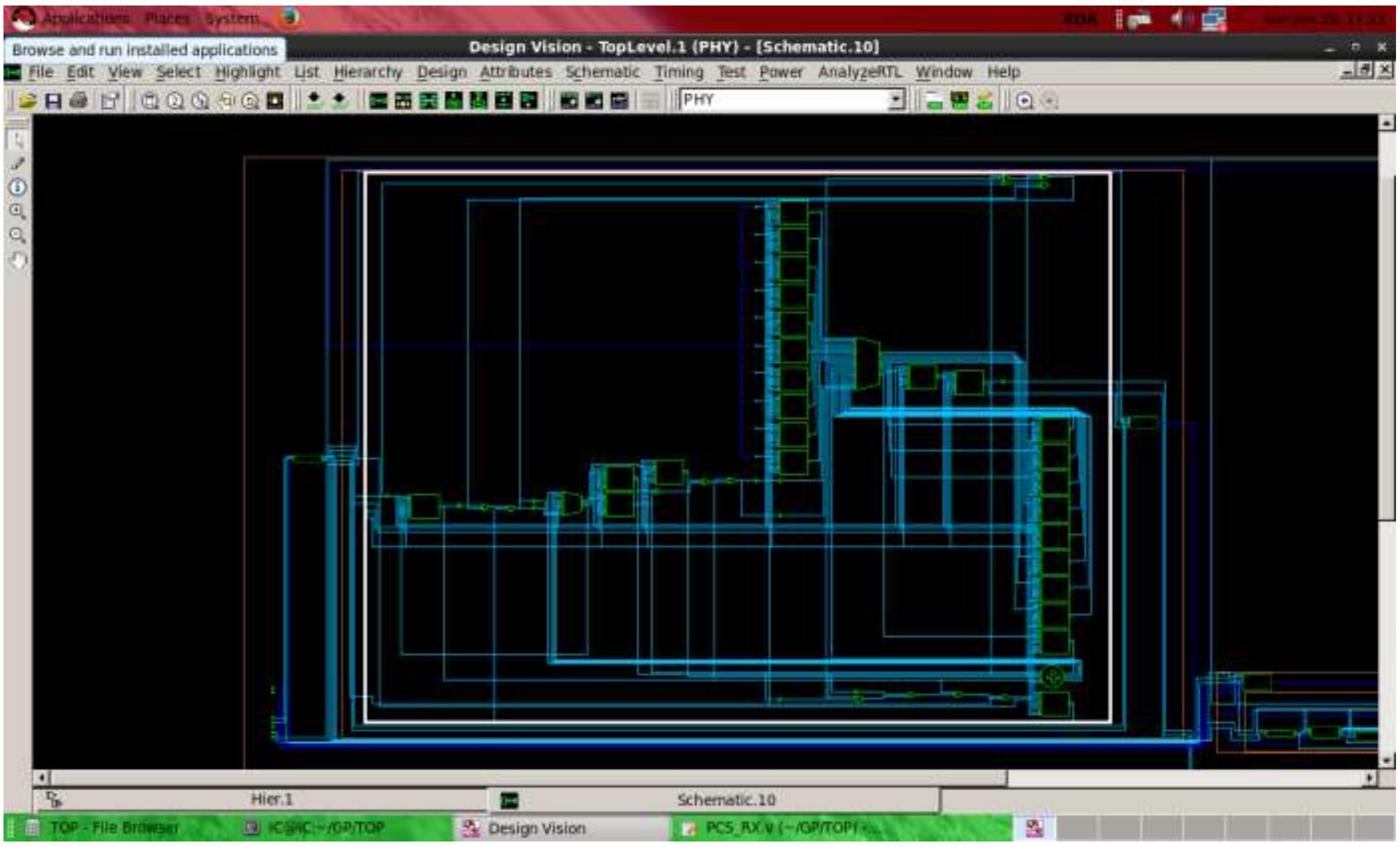

Applications Places System
Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.10
TOP - File Browser
ICBRC:~/GP/TOP
Design Vision
PCS_RX.v (~/GP/TOP) -

Figure 12 PMA TX

PMA RX Synthesis

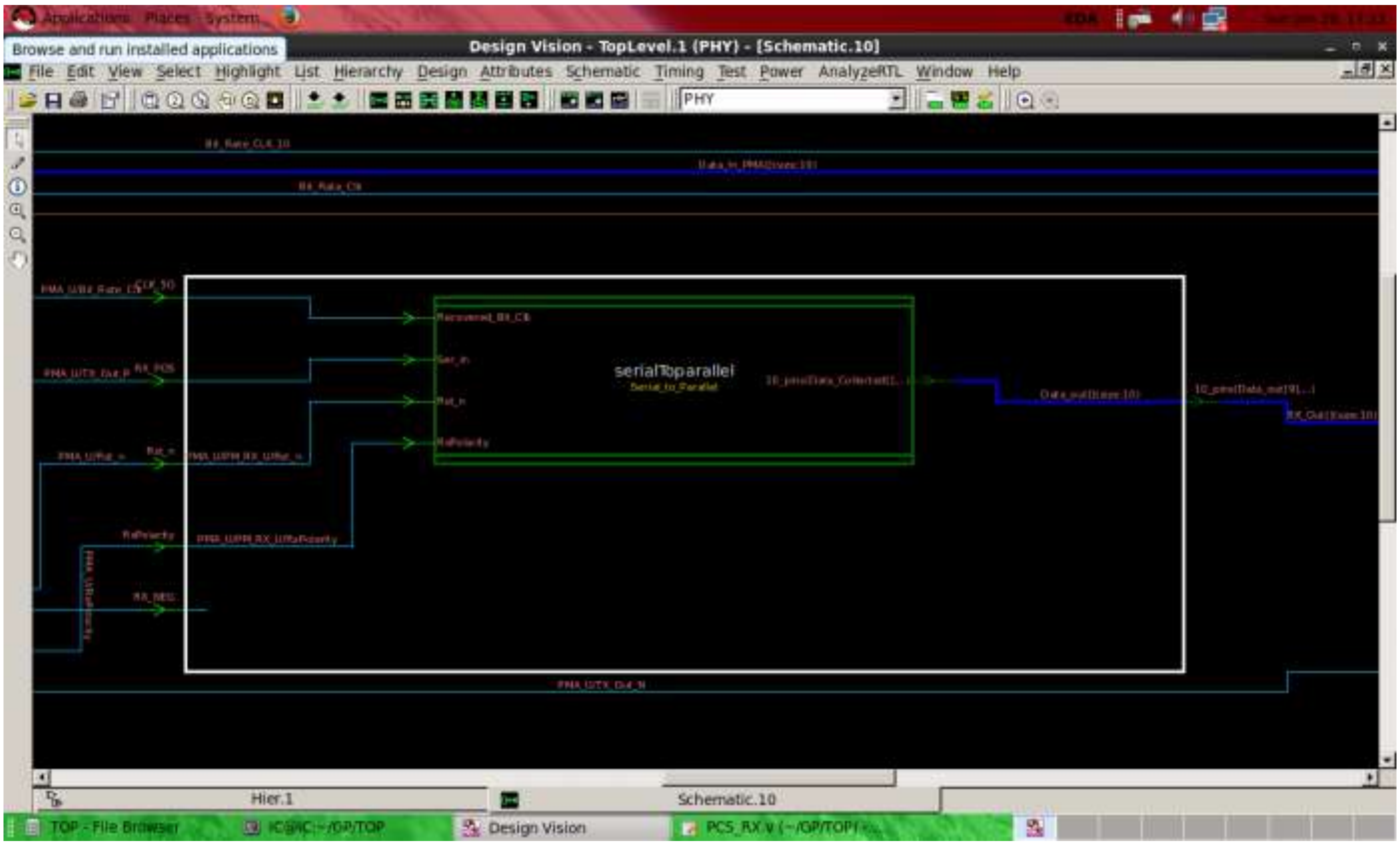

Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
serialToparallel
Serial_to_Parallel
Hier.1
Schematic.10
Design Vision

Figure 13 PMA RX

Common Block

The Common Block contains the PLL(Phase Locked loop) which is utilized in the design of USB PHYs (Physical Layers). The PLL is a crucial component that helps in generating stable clock signals, ensuring proper synchronization, and meeting the timing requirements of USB communication.

The PLL is responsible for generating stable and precise clock signals used in USB communication

The PLL include a clock multiplication function, allowing the USB PHY to generate higher-frequency clocks derived from a lower-frequency reference clock. Which is useful for achieving high-speed data rates.

In the design, the PLL takes as input Ref clock which is 100MHz and generates from it clock with faster rate:

- the serial clock which is 5GHz(Bit_rate_clk)

the fast clock is then given to a clock divider and according to the Div_ratio it is reduced to a lower speed:

- the word clock which is the time for 10bits (Bit_Rate_clk_10)
- the PCLK which changes according to the Bus width so using the 32 bus width takes 4 word clock period,16 bus width takes 2 word clock period, 8 bus width is equal

Common block "top module"

Name	Direction	Width	Description
Ref_Clk	Input	1	Reference Clk (100 MHz)
DataBusWidth	Input	6	Data Width (8,16,32)
Rst_n	Input	1	Active Low Reset
Bit_Rate_Clk	Output	1	Serial CLK(5G)
Bit_Rate_Clk_10	Output	1	WordClk
PCLK	Output	1	Parallel Clk

PLL

Name	Direction	Width	Description
Ref CLK	Input	1	100 MHz clk to generate high rate clks
CLK	Input	1	5 giga clock

Clock Divider

Name	Direction	Width	Description
Ref_CLK	Input	1	Serial CLK(5G)
Div_ratio	Input	8	Ratio for dividing high rates to low one
Rst_n	Input	1	Active Low Reset
Divided clk	Output	10	Low rate clk

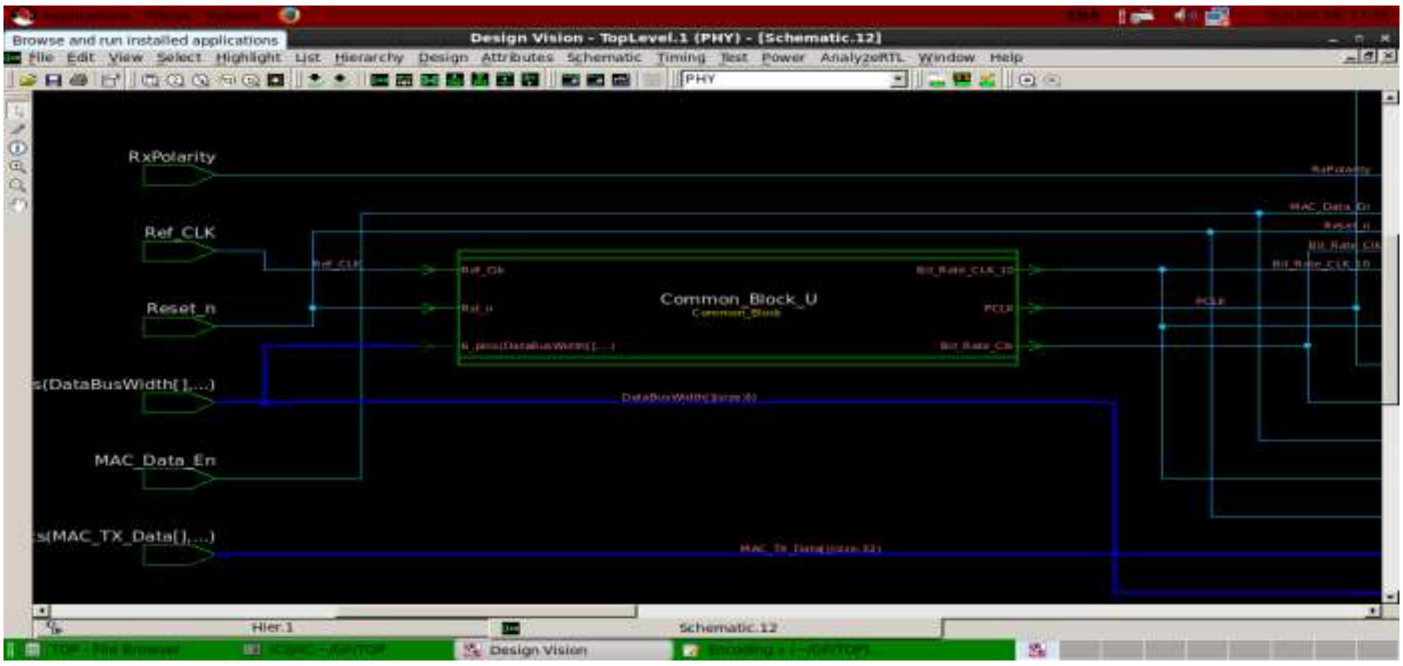

Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
RxPolarity
Ref_CLK
Reset_n
s(DataBusWidth[],...)
MAC_Data_En
s(MAC_TX_Data[],...)
Common_Block_U
Common_Block
PCLK
Hier.1
Schematic.12
Design Vision

Figure 14 Common Block

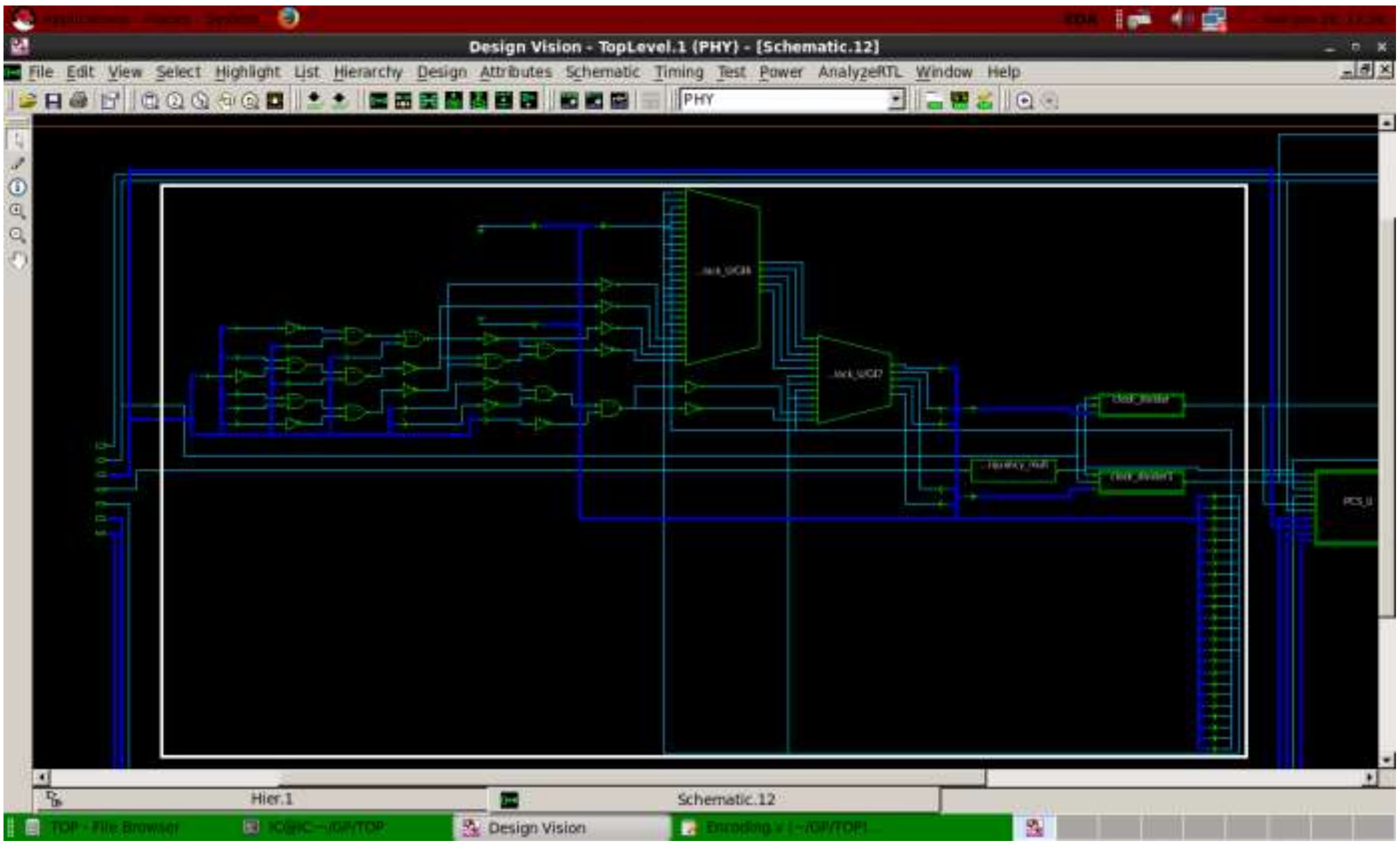

Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
clock_divider
clock_divider1
PCS_0
Hier.1
Schematic.12
Design Vision

Figure 15 inside Common block

Transmitter Blocks

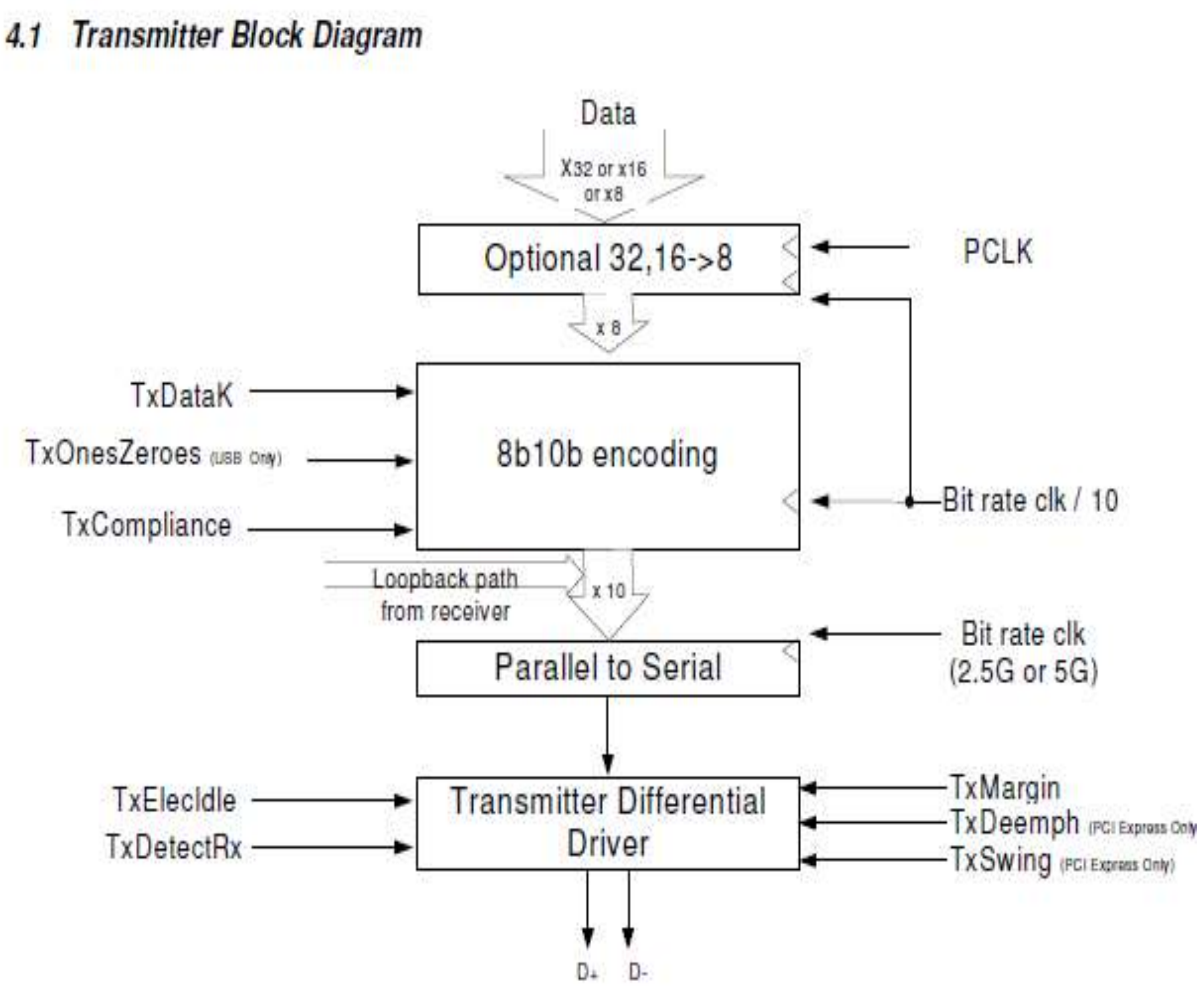


Figure 16 Transmitter Blocks

PCS TX

Gasket TX :

Data can be input to the module with different width 8,16,32

However for it to be transmitted it is divided into blocks of 8bits to be encoded to 10 bits and then sent to the receiver So the gasket module is responsible for diving the input into 8bits block to be transmitted every word clock cycle so within the period of the width all the 8-bit blocks are transmitted

Name	Direction	Width	Description
PCLK	Input	1	Parallel CLK
Bit_Rate_CLK_10	Input	1	Word CLK
Reset_n	Input	1	Active Low Reset
MAC_TX_Data	Input	32	Parallel Data sent from MAC
MAC_Data_En	Input	1	Enable
MAC_TX_DataK	Input	4	Indication From MAC to detect data sent is command or not
DataBusWidth	Input	5	Data sent Width
TXDataK	Output	1	Flag for command data
TXData	Output	8	Data to be encoded

Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.10
Design Vision

Figure 17 Gasket Tx

Line 8b/10b Encoding

The 8b/10b line coding scheme is used to encode data for transmission over a communication channel. It's designed to ensure reliable data transmission and provide certain properties such as DC balance (equal number of 0s and 1s) and error detection.

10-Bit Symbols: In this scheme, data is encoded into 10-bit symbols. Each 10-bit symbol represents a combination of data and control information. This encoding is more efficient for transmission than using a straightforward binary representation

Notation "D05.2": The "D05.2" notation is a way to describe a specific 10-bit symbol within the 8b/10b scheme. Let's break it down:

- "D" indicates that it's a data character.
- "05" represents the first five bits of the data character in binary. In binary, "05" is "00101.“
- ".2" indicates that the last bit is "0" in binary.

One Byte Representation: While "D05.2" is a specific encoding within the 8b/10b scheme, it doesn't directly represent one byte of data. The 8b/10b scheme doesn't have a strict one-to-one mapping between its symbols and bytes of data. The scheme balances between 0s and 1s to ensure reliable transmission.

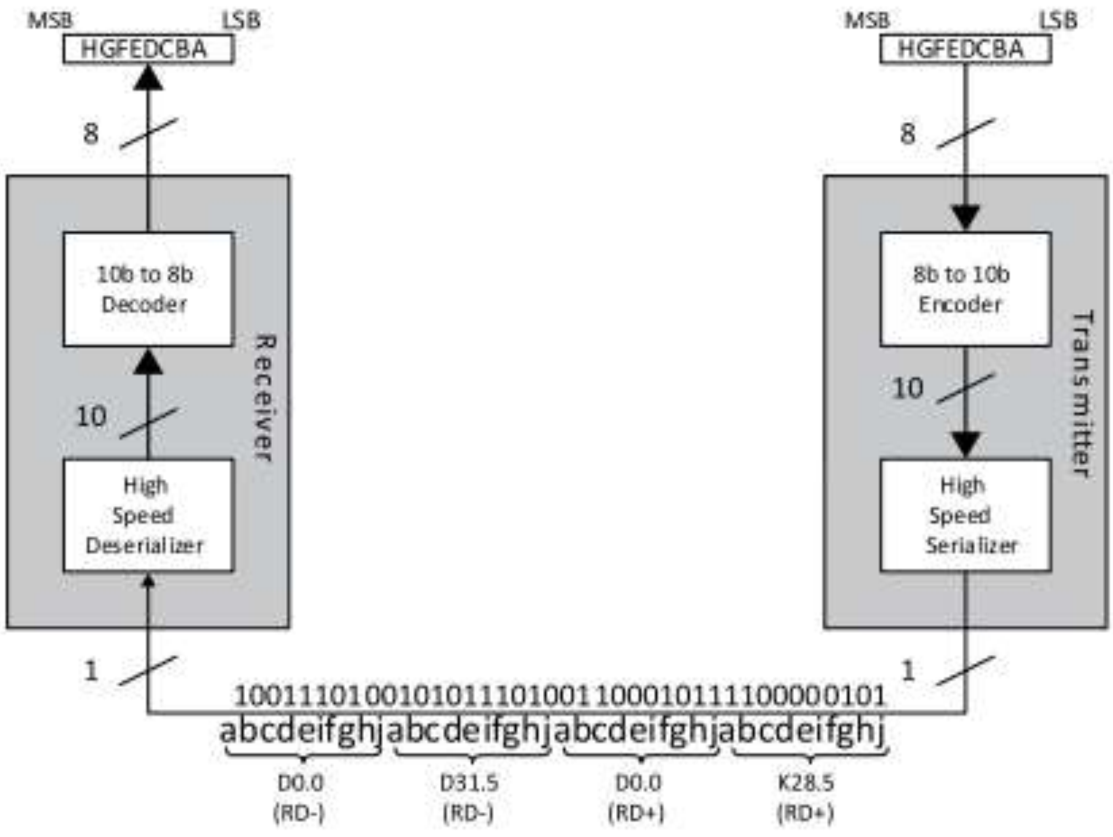


Figure 2.1. The 8b/10b Encoder/Decoder in a System

Figure 18 example1 of encoding,decoding

The Gen 1 PHY uses the 8b/10b transmission code. PHY transmits information using an adaptive 8B/10B code to bound the maximum run length of the code .

There are Two types of transmission characters Data and Special. Ordered Sets are known as combinations of transmission characters

MAC layer uses letter notation for describing information bits and control variable for example special symbol K28.5 here we just used letter and decimal numbers for describing not binary notation which we will explain in the following.

Information bit can hold only zero or one as value , control variable also can hold either the value D means “valid data byte” or the value K means “special code”

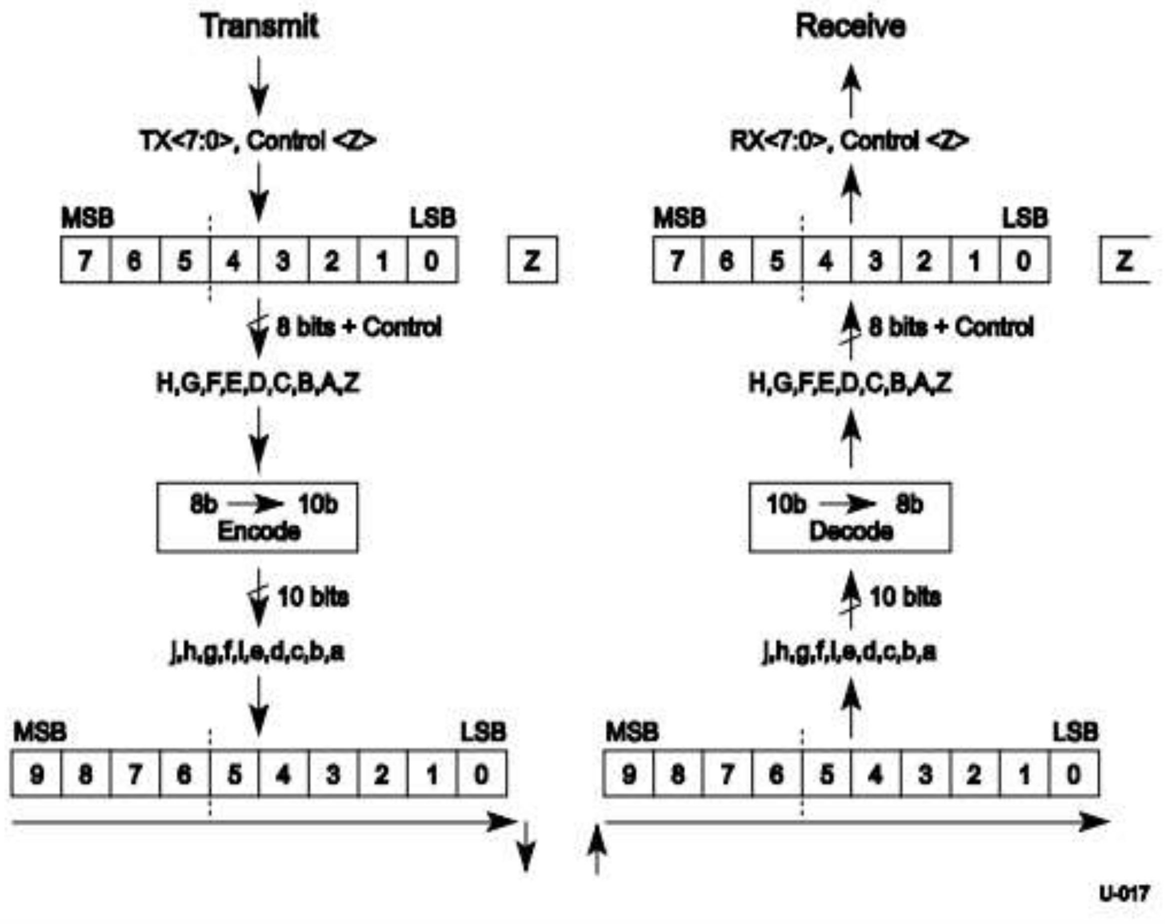
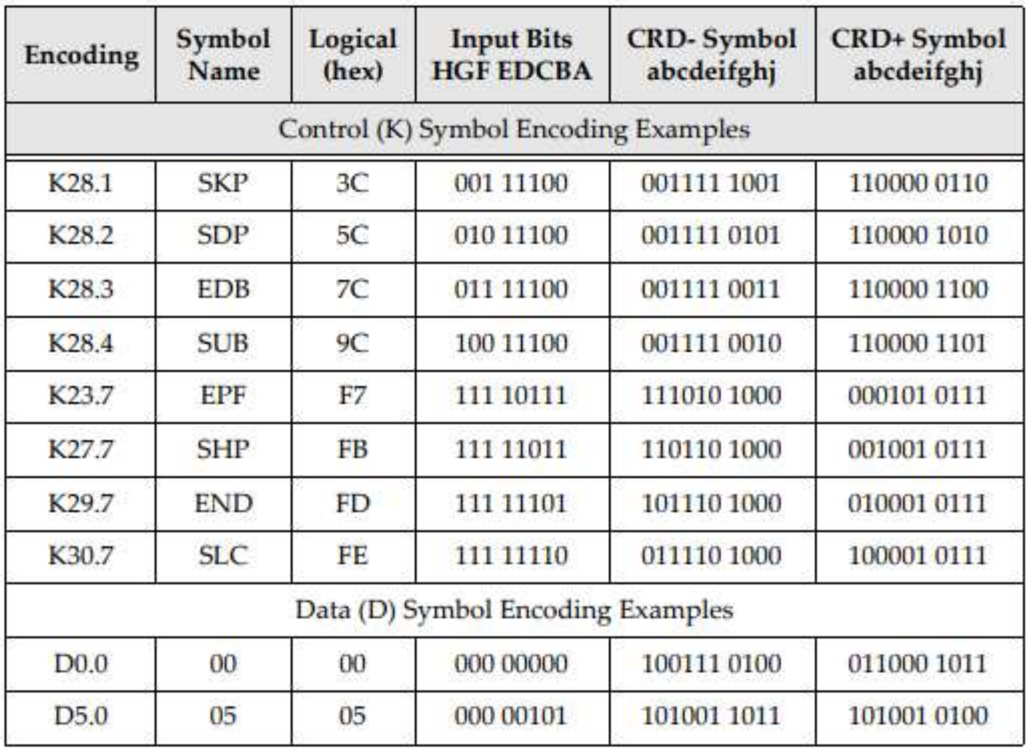


Figure 19 MAC layer letter notation with encode, decode

- Each 10-bit encoded symbol contains six 1’s and four 0’s, six 0’s and four 1’s, or five 1’s and five 0’s. Symbol encodings with more than six bits of one polarity are not valid.
- The D/K bit accompanying each byte into the encoder assures that control (K) symbols and data (D) symbols are unique sets. Even if a data byte and control byte are the same numerically, their symbol encodings are different.

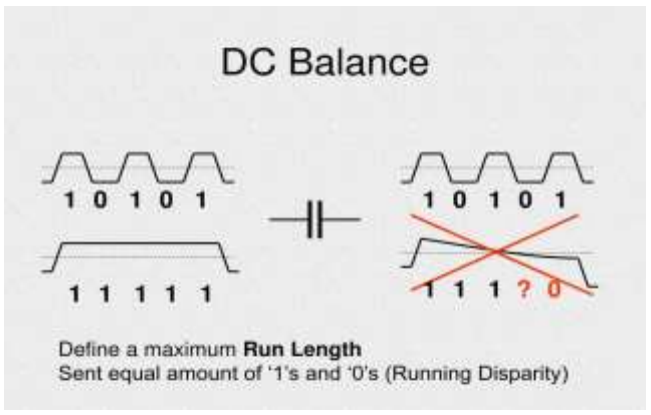


Encoding	Symbol Name	Logical (hex)	Input Bits HGF EDCBA	CRD- Symbol abcdeifghj	CRD+ Symbol abcdeifghj
Control (K) Symbol Encoding Examples					
K28.1	SKP	3C	001 11100	001111 1001	110000 0110
K28.2	SDP	5C	010 11100	001111 0101	110000 1010
K28.3	EDB	7C	011 11100	001111 0011	110000 1100
K28.4	SUB	9C	100 11100	001111 0010	110000 1101
K23.7	EPF	F7	111 10111	111010 1000	000101 0111
K27.7	SHP	FB	111 11011	110110 1000	001001 0111
K29.7	END	FD	111 11101	101110 1000	010001 0111
K30.7	SLC	FE	111 11110	011110 1000	100001 0111
Data (D) Symbol Encoding Examples					
D0.0	00	00	000 00000	100111 0100	011000 1011
D5.0	05	05	000 00101	101001 1011	101001 0100

Figure 20 sample of encoded data and special character

DC Balance and run length

A DC-balanced serial data stream means that it has the same number of 0s and 1s for a given length of data stream. DC-balance is important for certain media as it avoids a charge being built up.


DC Balance
1 0 1 0 1
1 1 1 1 1
1 0 1 0 1
1 1 1 ? 0
Define a maximum Run Length
Sent equal amount of '1's and '0's (Running Disparity)

The run-length is defined as the maximum numbers of contiguous 0s or 1s in the serial data stream. A small run length data stream provides data transitions within a small length of data. Data transitions are essential for clock recovery.

The PLL of the CDR generates a phase-adjustable output clock from the reference clock input. Transitions on the serial data stream provide the transmission clock phase information to the PLL and allow the PLL to recover the transmission clock with the correct phase. Note that the reference clock input is always necessary for the CDR. The serial data stream embeds the phase of the transmission clock, not the clock itself. This reference clock comes from the receiver system, not the transmitter system.

Code Mapping

The coding scheme breaks the original 8-bit data into two blocks, 3 most significant bits (y) and 5 least significant bits (x). From the most significant bit to the least significant bit, they are named as H, G, F and E, D, C, B, A. The 3-bit block is encoded into 4 bits named j, h, g, f. The 5-bit block is encoded into 6 bits named i, e, d, c, b, a. As seen in Figure ., the 4-bit and 6-bit blocks are then combined into a 10-bit encoded value.

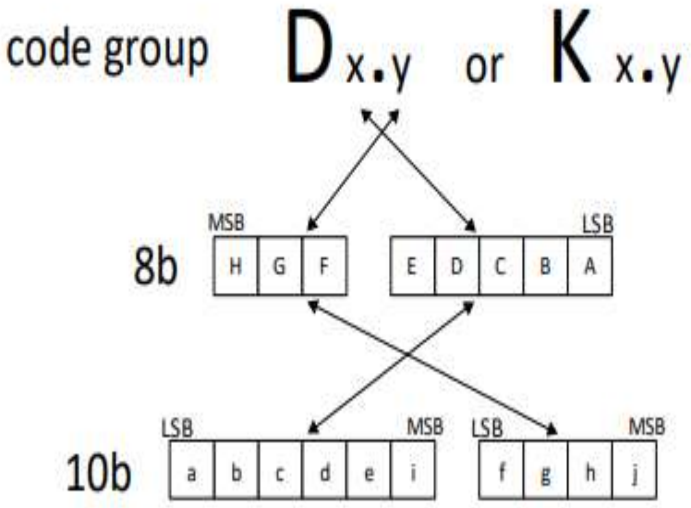
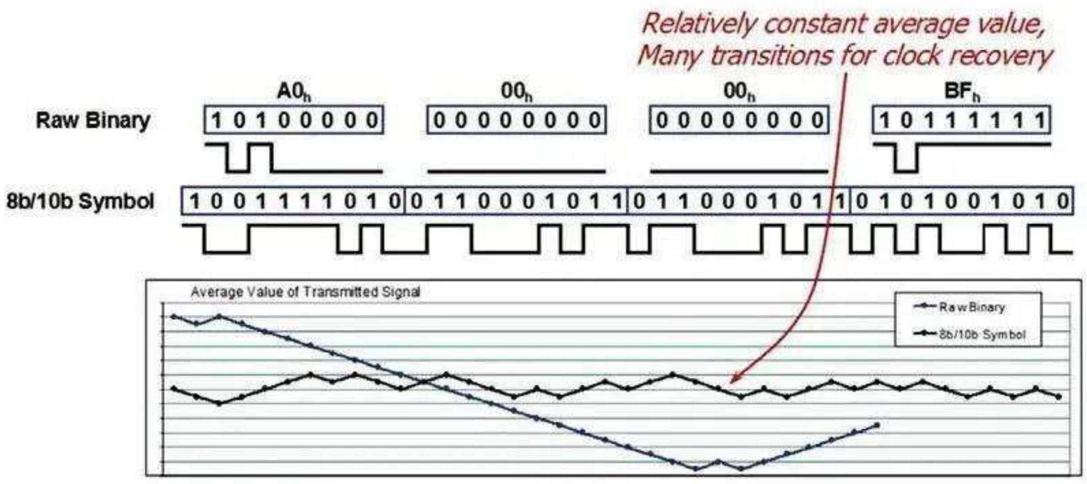
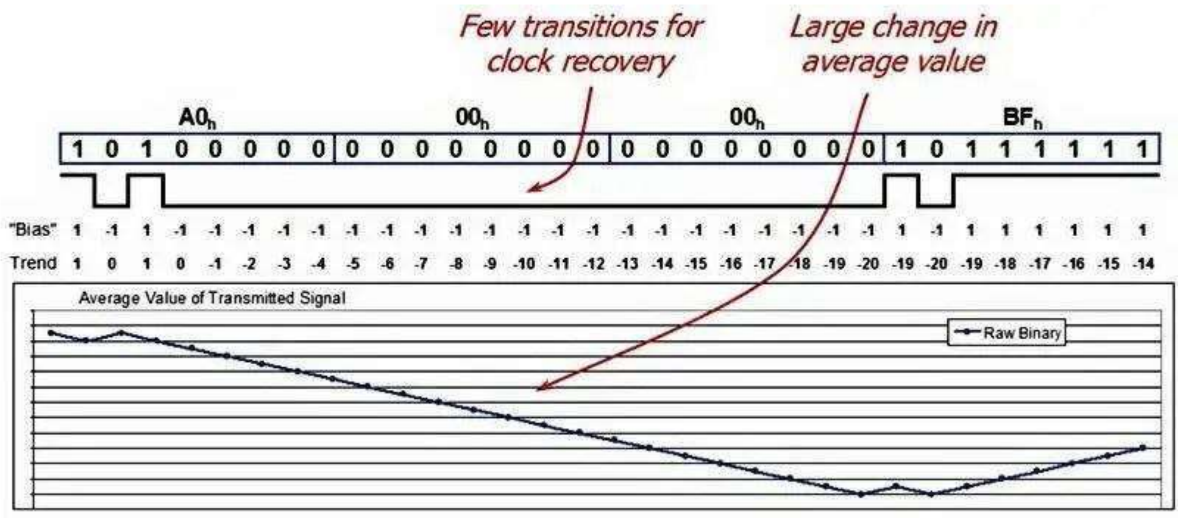


Figure 21 code mapping

Running Disparity

- the 8b/10b encoding scheme actually supports two possible encoded values for each data and control byte. This is because transmitters are required to assure that outbound serial data, over time, contains an equal (balanced) number of 0's and 1's. The on-going difference in the number of transmitted bits of each polarity is referred to as the running disparity. Tracking the current running disparity (CRD) and correcting any imbalance is critical on the AC-coupled SuperSpeed link and one of the important motivations for 8b/10b encoding..
- each symbol out of the 8b/10b encoder has one of the following 10-bit properties:
  - It is comprised of six 1's and four 0's (a positive disparity) OR
  - It is comprised of six 0's and four 1's (a negative disparity) OR
  - It is comprised of five 1's and five 0's (a neutral disparity)





Encoder "Top Block"

Name	Direction	Width	Description
MAC_Data_En	Input	1	Enable the encoder to do
TXDataK	Input	1	Define whether data is comman or actual data
Data	Input	8	Data from MAC
Bit_Rate_10	Input	1	Clk to send 10-bits to PMA
Rst	Input	1	Active low reset
Data_Out	Output	10	Data sent to PMA

Line_encoding

Name	Direction	Width	Description
Enable	Input	1	Enable the encoder to do
TXDataK	Input	1	Define whether data is comman or actual data
Data	Input	8	Data to be encoded
Encoded_data_pos	Output	10	Encoded data
Encoded_data_neg	Output	10	Encoded data

FSM_RD ”Running Disparity”

Name	Direction	Width	Description
Enable	Input	1	Enable the encoder to do
TXDataK	Input	1	Define whether data is comman or actual data
Data_neg	Input	10	RD(-) Data
Data_pos	Input	10	RD(+) Data
Bit_Rate_10	Input	1	Clk to send 10-bits to PMA
Rst	Input	1	Active low reset
Data_10	Output	10	Current Running Disparity Data

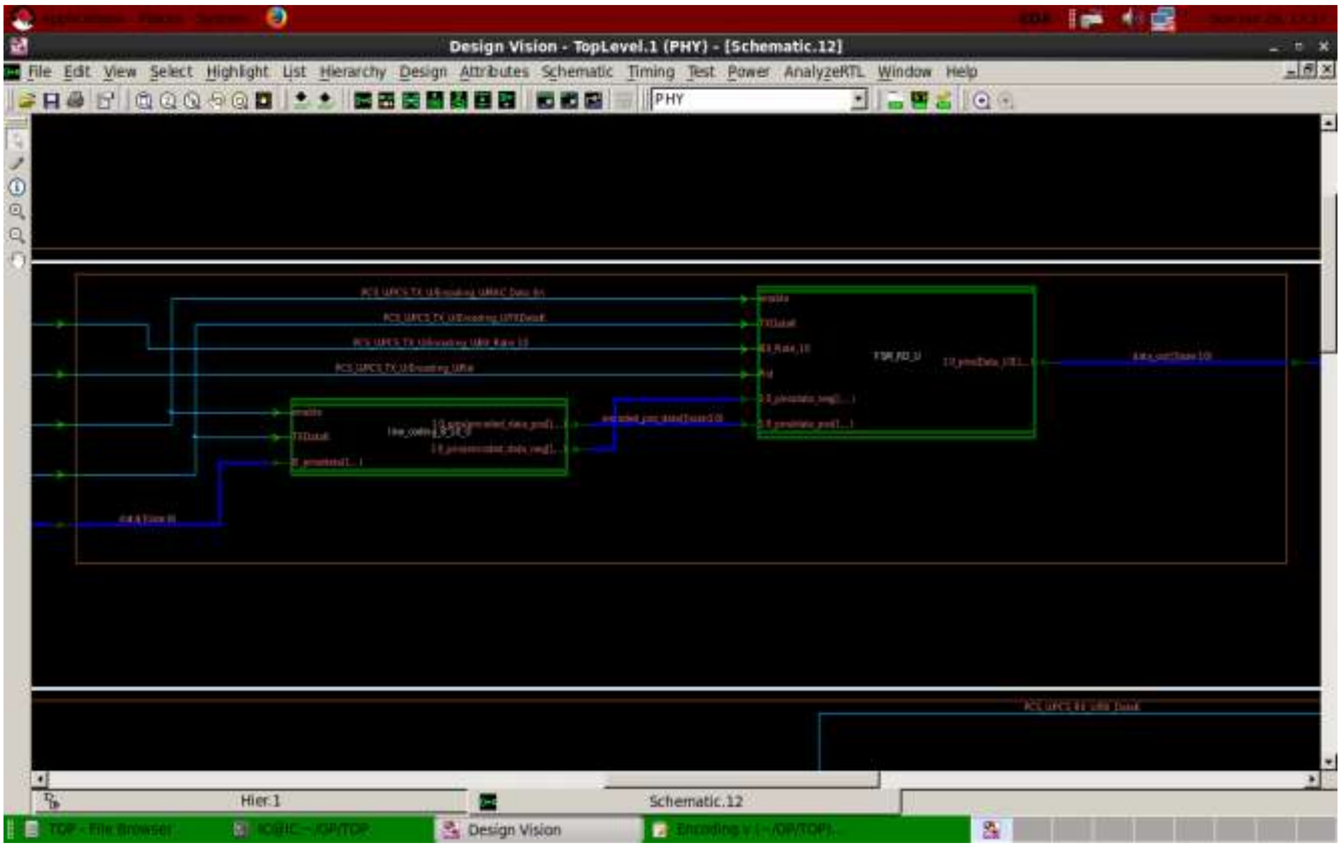

Design Vision - TopLevel.1 (PHY) - [Schematic.12]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
enable
TXDataK
Bit_Rate_10
Rst
FSM_RD_U
Hier.1
Schematic.12
Design Vision

Figure 22 line encoding

PMA TX

Parallel to serial:

USB(universal serial bus) transmits data to receiver serial as transmitting data serially instead of parallel have many advantages such as

- reduced wiring complexity as transmitting data in parallel requires multiple wires unlike sending data serially
- Improving signal integrity as Parallel communication over multiple wires can lead to issues such as signal skew, where bits arrive at slightly different times due to variations in wire lengths, while serial communication, being a single stream of bits, is less susceptible to these timing discrepancies, resulting in better signal integrity.
- Longer Transmission Distances: Serial communication is often more suitable for long-distance communication as The use of differential signaling (such as in USB) helps reduce the impact of noise and interference over longer cable lengths.

The parallel data is received with relatively slow speed and the transmitted with faster speed where the serial data rate is 10 times faster than the parallel data rate

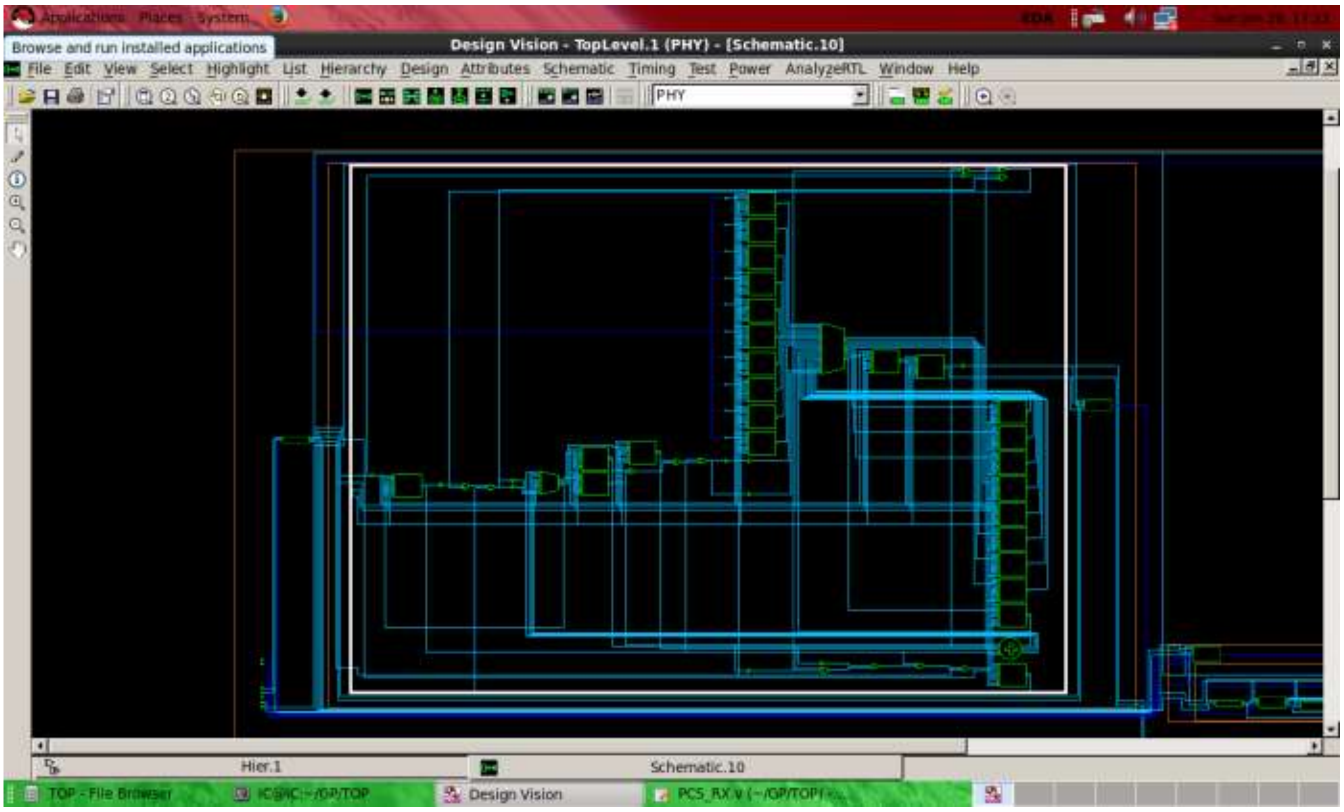

Applications Places System
Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.10
TOP - File Browser
ICGP/IC:~/GP/TOP
Design Vision
PCS_RX.v (~/GP/TOP/...

Figure 23 parallel to serial

Receiver Blocks

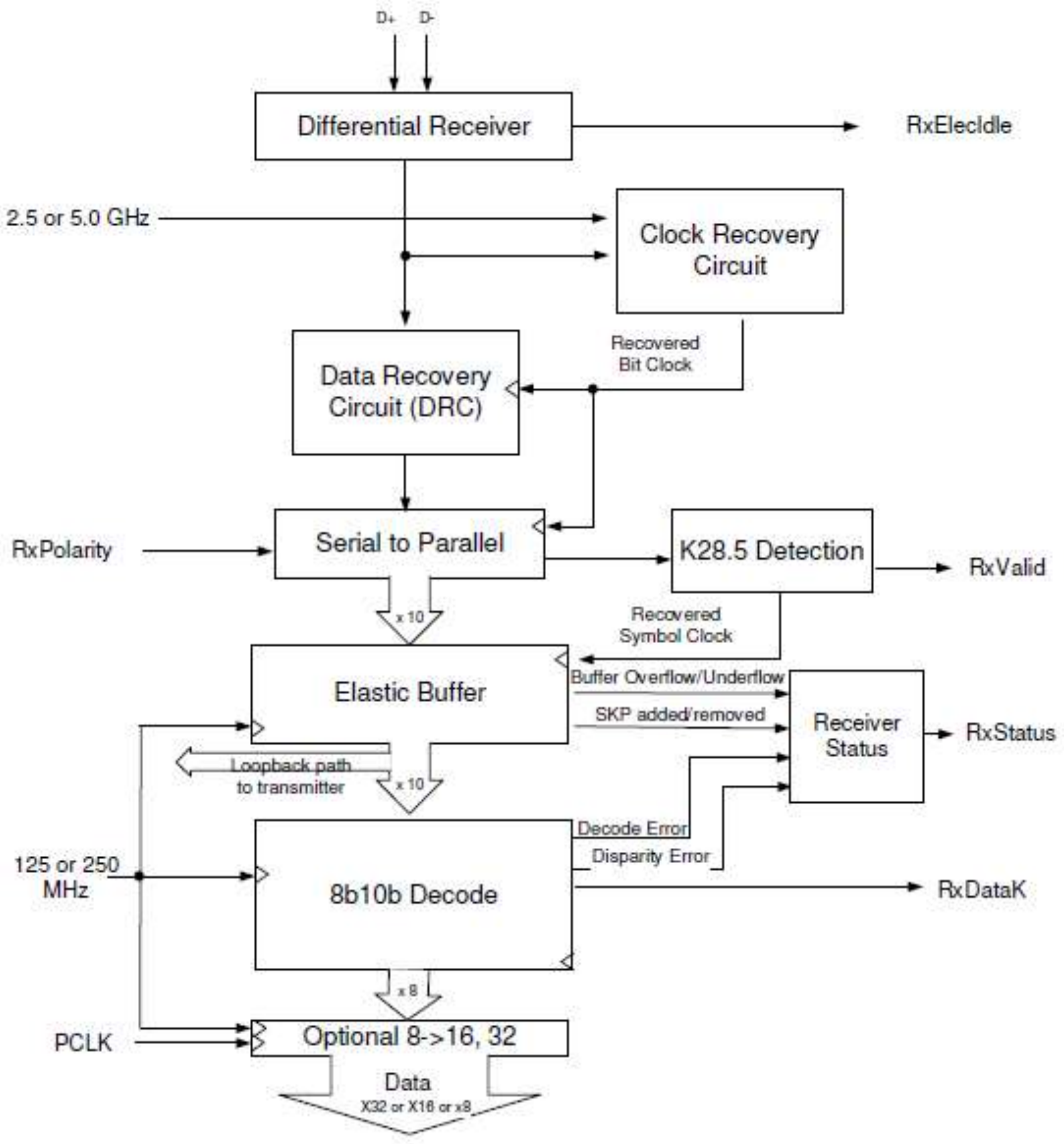
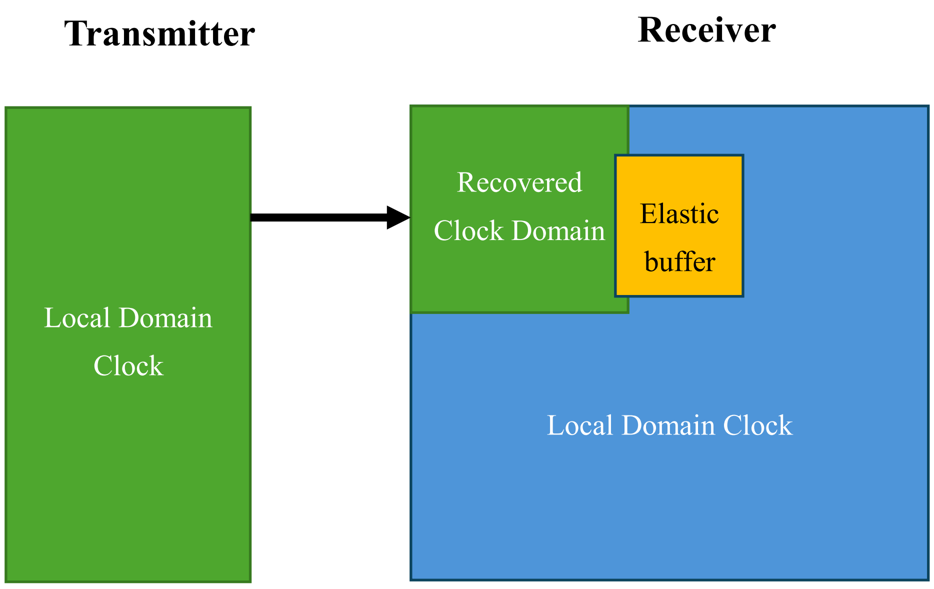


Figure 24 Receiver Blocks

PCS RX

Elastic Buffer:

Introduction



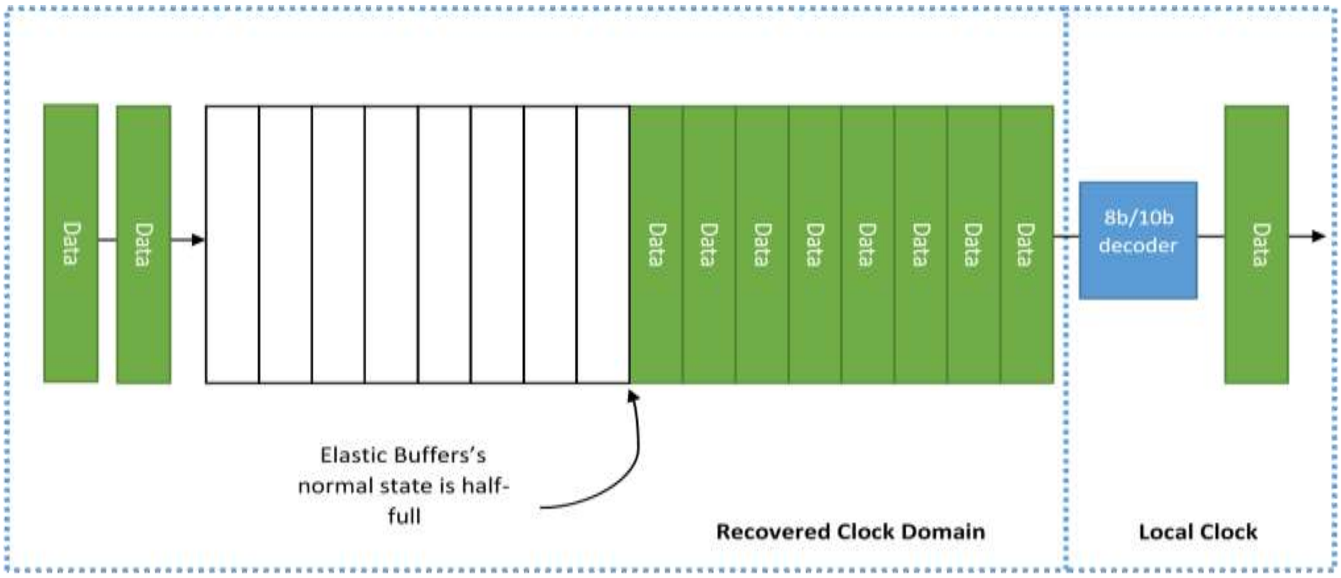
Elastic Buffers (also known as Elasticity Buffers, Synchronization Buffers, and Elastic Stores) are used to ensure data integrity when bridging two different clock domains. This buffer is simply a FIFO (First-In-First-Out) where data is deposited at a certain rate based on one clock and removed at a rate derived from a different clock. Because these two clocks could (and almost always do) have minor frequency differences, there is the potential for this FIFO to eventually overflow or underflow. To avoid this situation, an Elastic Buffer has the ability to insert or remove special symbols, during specified intervals that allow the buffer to compensate for the clock differences.

The extreme clock difference happens when TX clock is -300ppm, the RX clock is -300ppm and the spread Spectrum clocking(SSC) is -5000ppm, the total mismatch is 5600ppm.so every 178 symbo1 (1M/5600) the transmitter will fall behind with one symbol. If not addressed the receiver will encounter data underflow (no symbols to process)

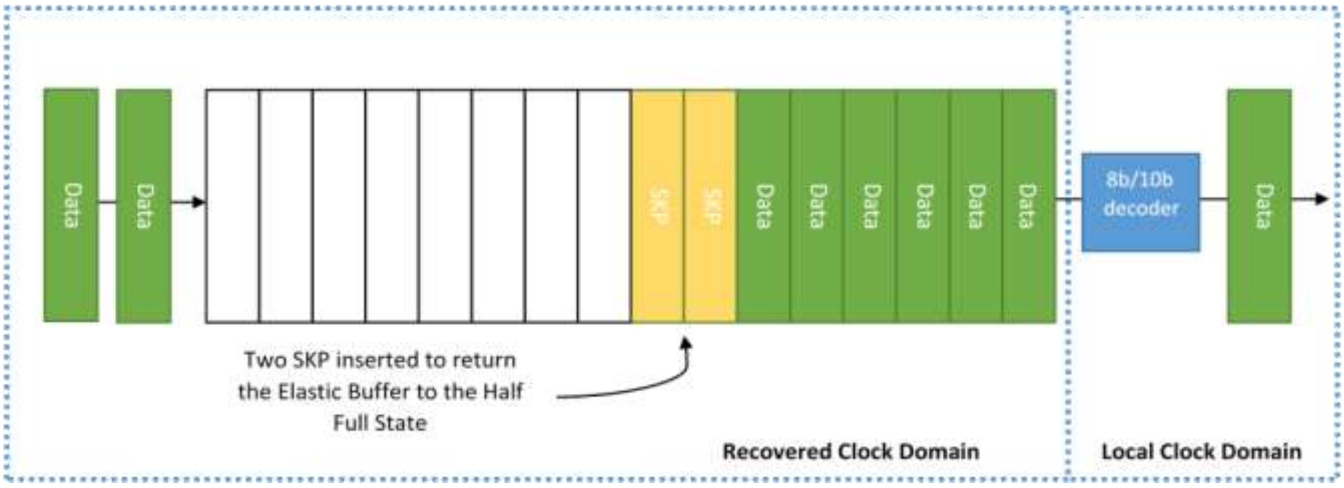
The other extreme clock difference happens when TX clock is +300ppm, the RX clock is -300ppm and the spread Spectrum clocking(SSC) is 0ppm, the total mismatch is 600ppm.so every 1666 symbo1 (1M/600) the transmitter will have sent one extra symbol. If not addressed the receiver will encounter data overflow(a dropped symbol during processing)

The elastic buffer is a crucial to handle this mismatch and to avoid the timing errors between the read and write operations which may lead to data loss. The elastic buffer is mainly an asynchronous FIFO with additional functionality. The elastic buffer is used to deal with SKP to avoid losing data. Normally the System clock and receive clock, which is used in the elastic buffer, should be the same which is 5G ,however there can be slight changes which ranges from -5300 ppm to 300ppm.

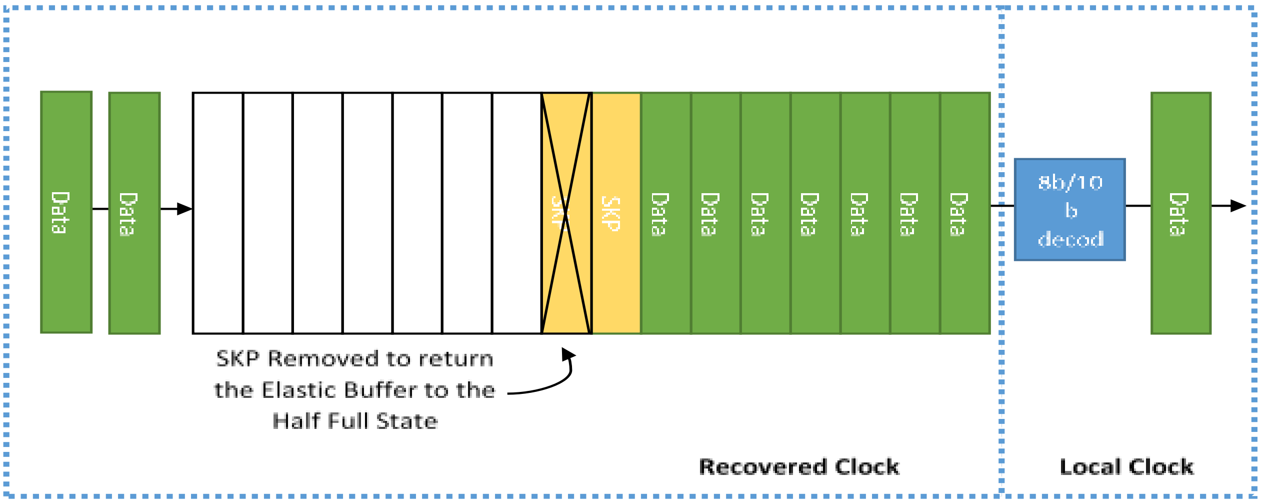
The elastic buffer implemented uses Nominal Half Full Buffer where the elastic buffer should always have the buffer half full and adding the SKP or removing it is done to keep the elastic buffer half full.



Adding skp to keep the buffer at nominal half full



Removing skp to keep the buffer at nominal half full



Elastic Buffer Design:

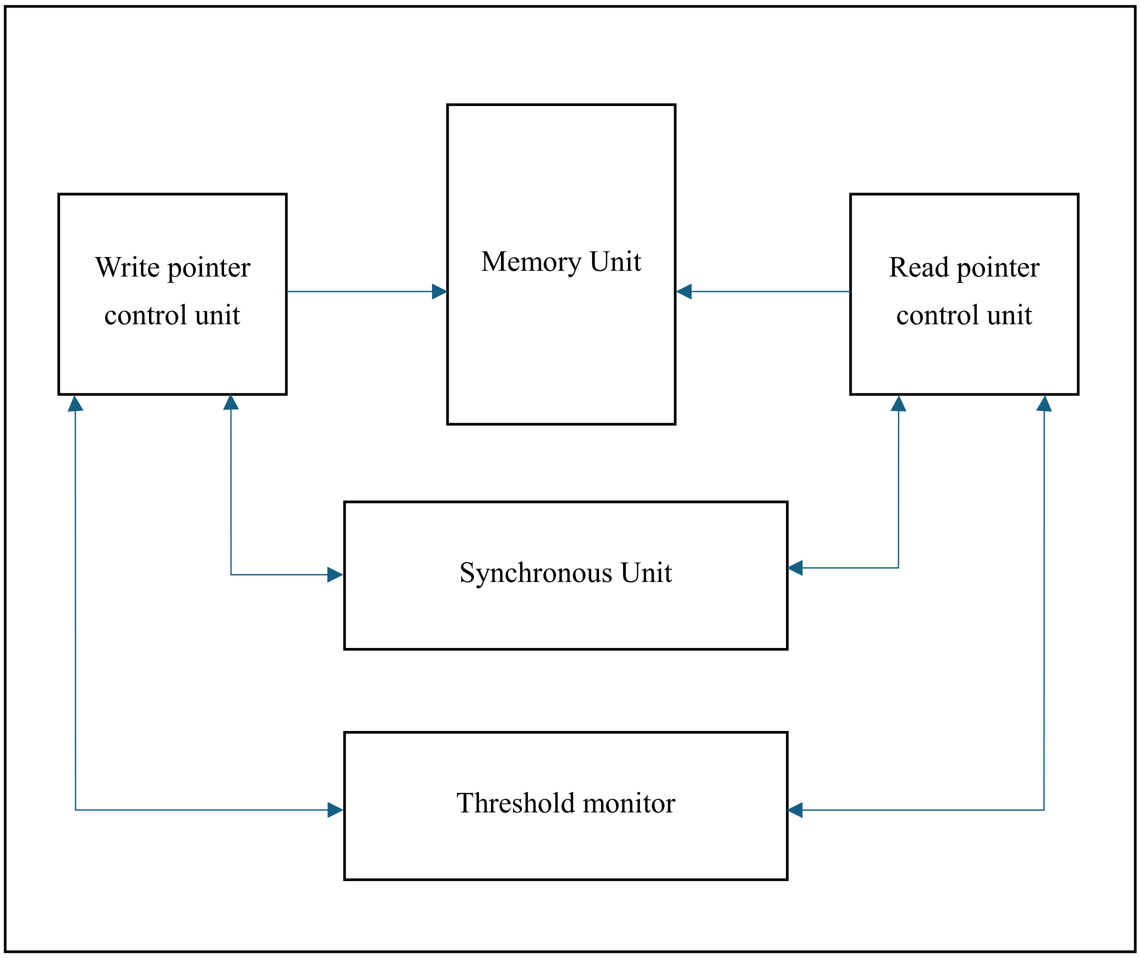


Figure 25 elastic buffer

The elastic buffer is mainly designed

- Memory unit
  It is mainly a FIFO where data is stored and read from it. The elements stored are the 10 bits data received which can be data or command like SKP. It determines the position to read from or write to using the read and write pointer which is received from other blocks.
- Write pointer control unit
  This module produces the binary and gray write pointer, which is used for the selection of the correct address of the memory unit while taking into consideration if a SKP needs to be deleted. It generates the overflow signal by comparing between the read and write address. . The gray code of read pointer is synchronized to the Recovered Clock Domain and compared with the gray code of the write pointer, then the unit produces the full Flag Read pointer control unit
- Read pointer control unit
  This module produces the binary and gray read pointer, which is needed to read from the correct address in the memory unit and it generates the request signal of SKP add after being compared with the write pointer. The gray code of write pointer is synchronized to the local Clock Domain and compared with the gray code of the read pointer, then the unit produces the empty Flag.

- Threshold unit

  This module is used to check whether the buffer exceeded the limit in comparison with the size of half the array. So if the number of elements bigger than 8 SKP remove request is made
- Synchronous Unit

  The function of this module is to synchronize the gray code of the write pointer and the read pointer so the read and write pointer checks on them and produce the empty and full flag

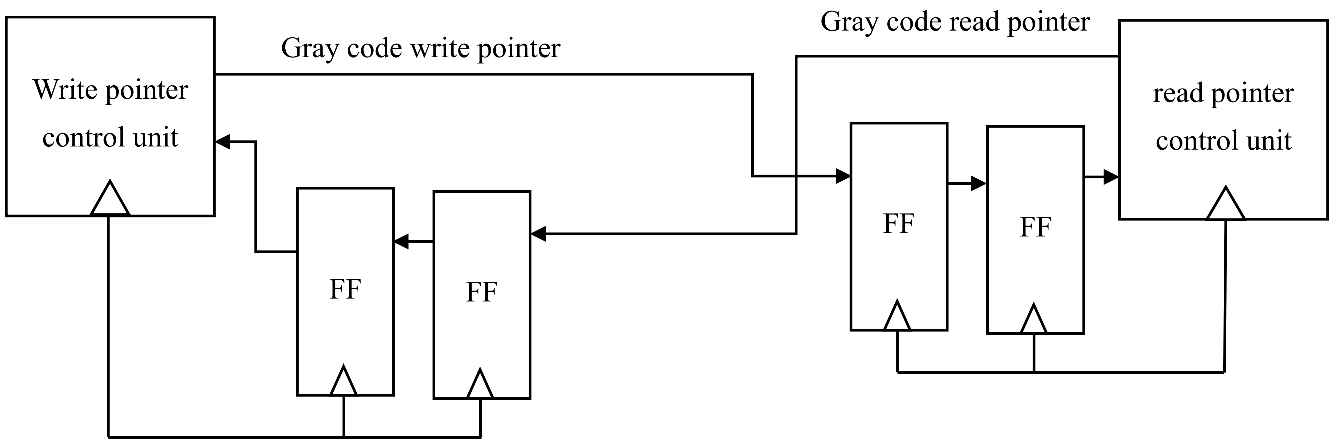


Figure 26 elastic buffer synchronous unit

Gray code is used as the change between a number and the following one is in only one bit so by using Gray code and synchronizers errors is avoided even when dealing with two clock domains, Recovered Clock Domain and Local Clock Domain.

Flow charts for SKP Delete and SKP add Operations

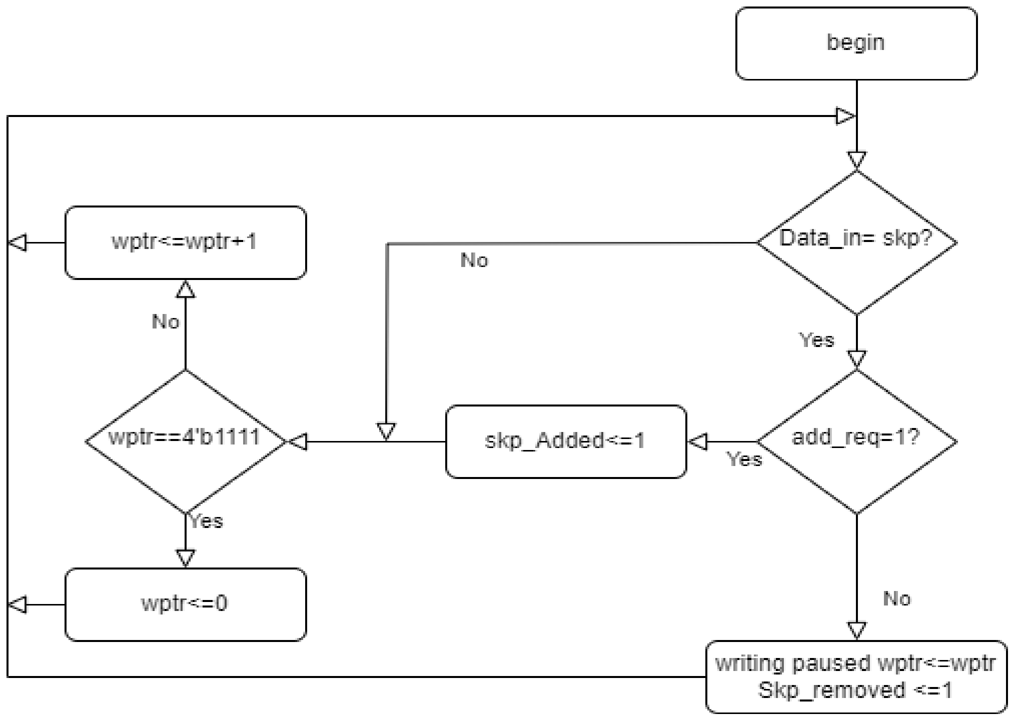


Figure 27 flow chart for SKP Handling

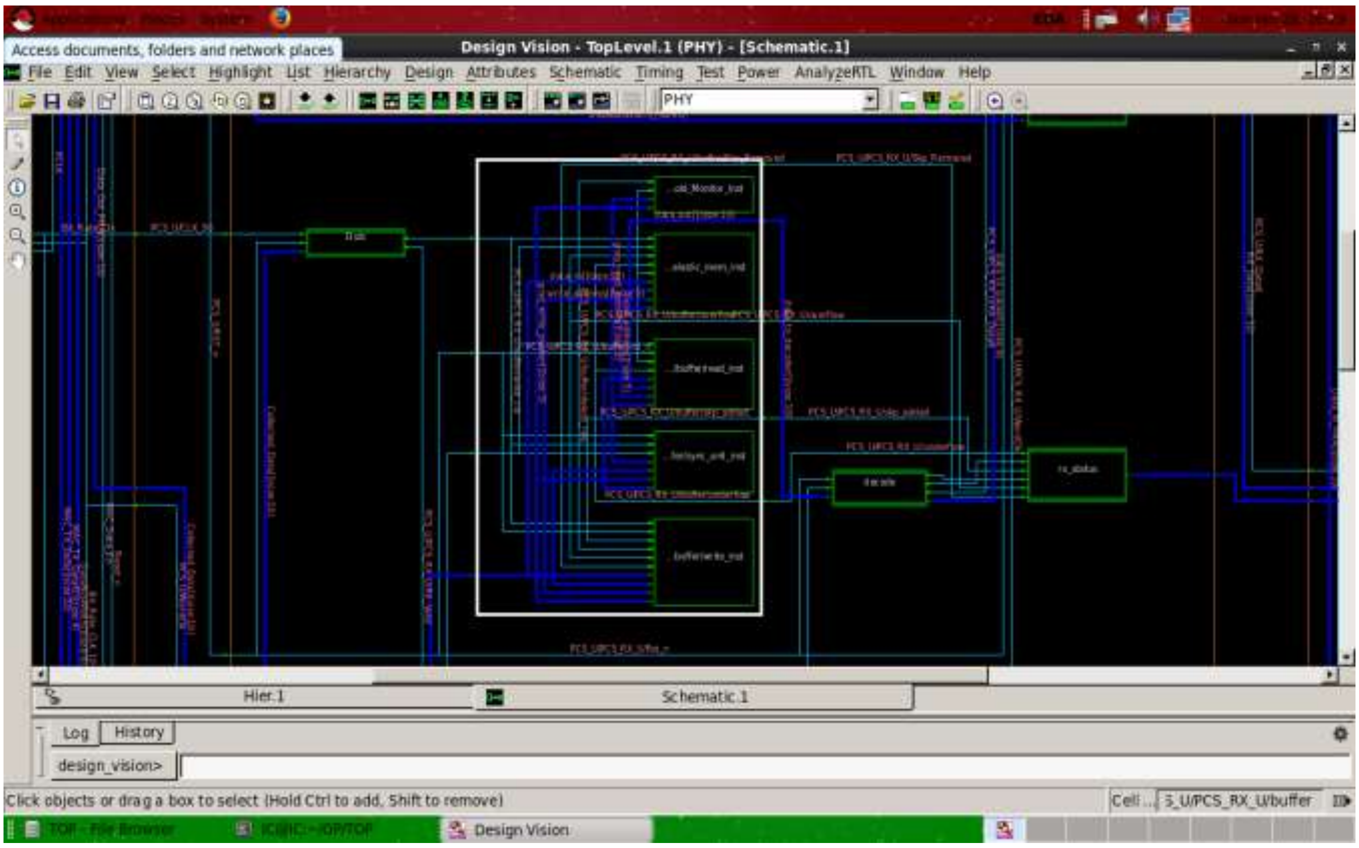
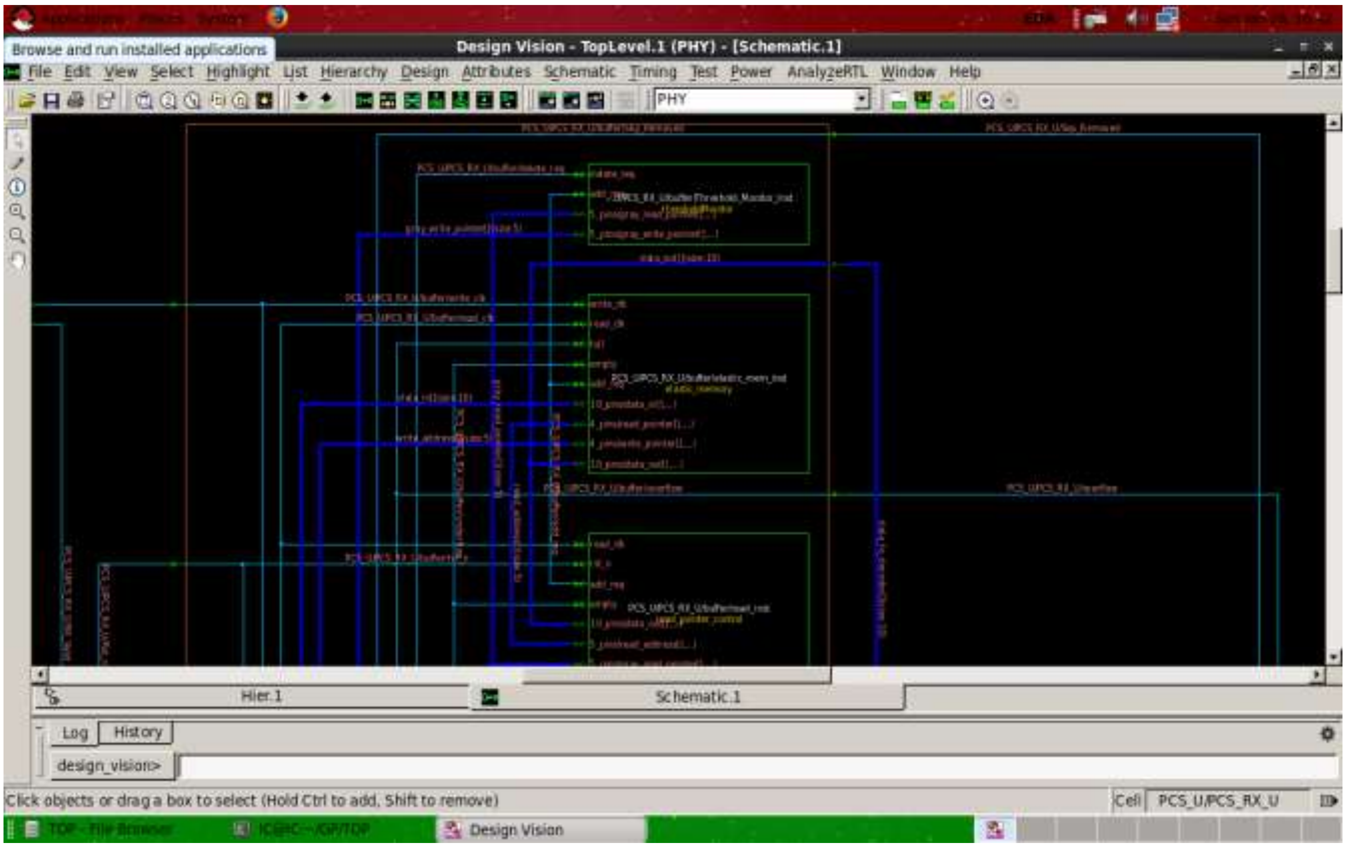
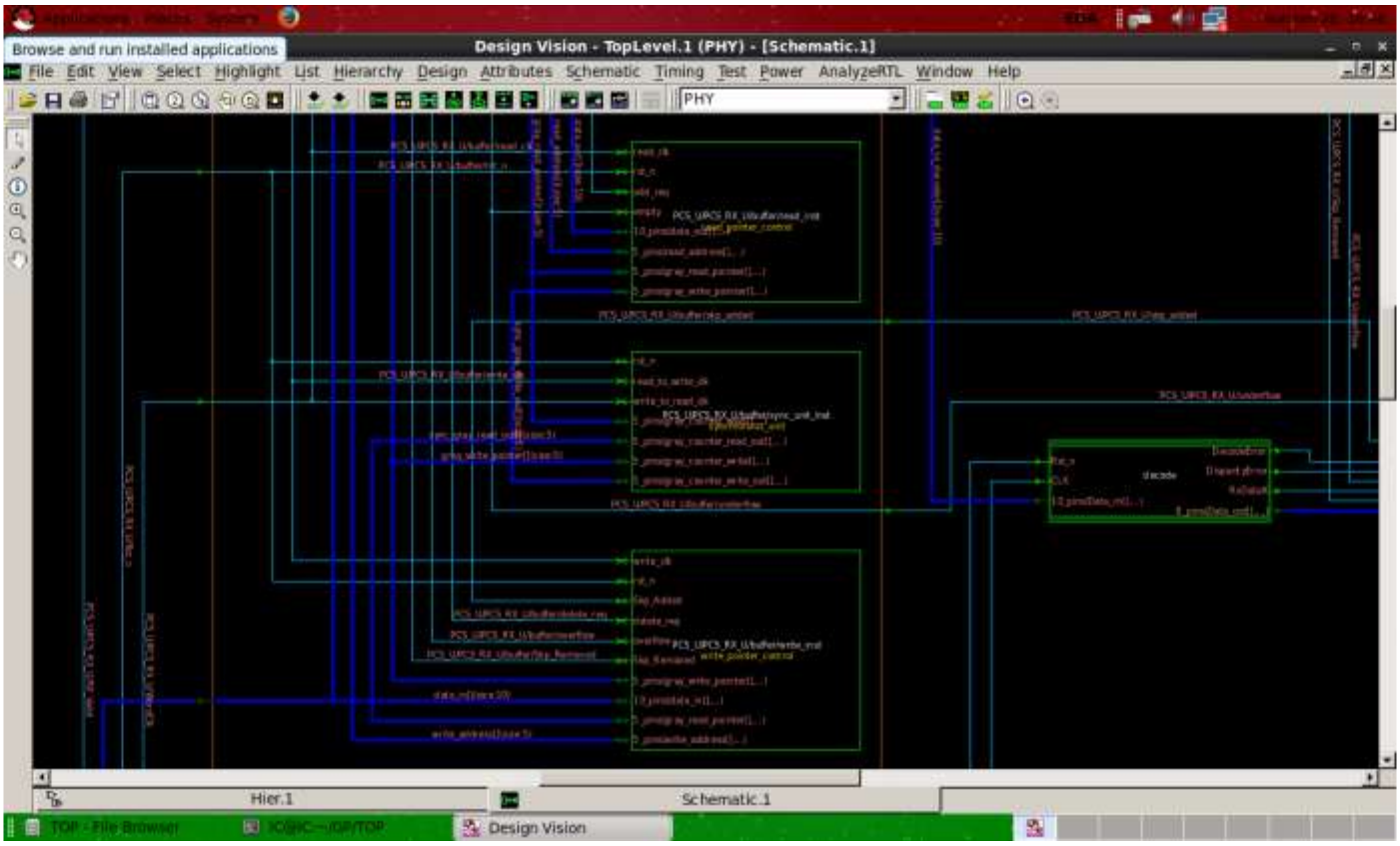

Access documents, folders and network places
Design Vision - TopLevel.1 (PHY) - [Schematic.1]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.1
Log History
design_vision>
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)
Cell ... S_U/PCS_RX_U/buffer
Design Vision

Figure 28 elastic buffer synthesis


Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.1]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.1
Log History
design_vision>
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove)
Cell PCS_U/PCS_RX_U
Design Vision


Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.1]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
PCS_UPCS_RX_U/buffer/read_inst
read_pointer_control
PCS_UPCS_RX_U/buffer/sync_unit_inst
PCS_UPCS_RX_U/buffer/write_inst
write_pointer_control
decode
DecodeError
DisparityError
RxDataK
Hier.1
Schematic.1
TOP - File Browser
Design Vision

Receiver Status:

This module gives feedback to the MAC with the current status according to the input flags such as overflow, underflow, disparity error and decode error

[2]	[1]	[0]	Description
0	0	0	Received data ok
0	0	1	USB superSpeed Mode: 1Skp ordered set added
0	1	0	USB superSpeed Mode: 1Skp ordered set removed
0	1	1	Reciever Detected
1	0	0	Both 8b/10b decode error and (optionally) receive disparity error
1	0	1	Elastic Buffer Overflow
1	1	0	Elastic Buffer Underflow this is unused if the elastic buffer is operating in nominal buffer empty mode
1	1	1	Receive disparity error (Reserved if Receive Disparity error is reported with code 0b100)

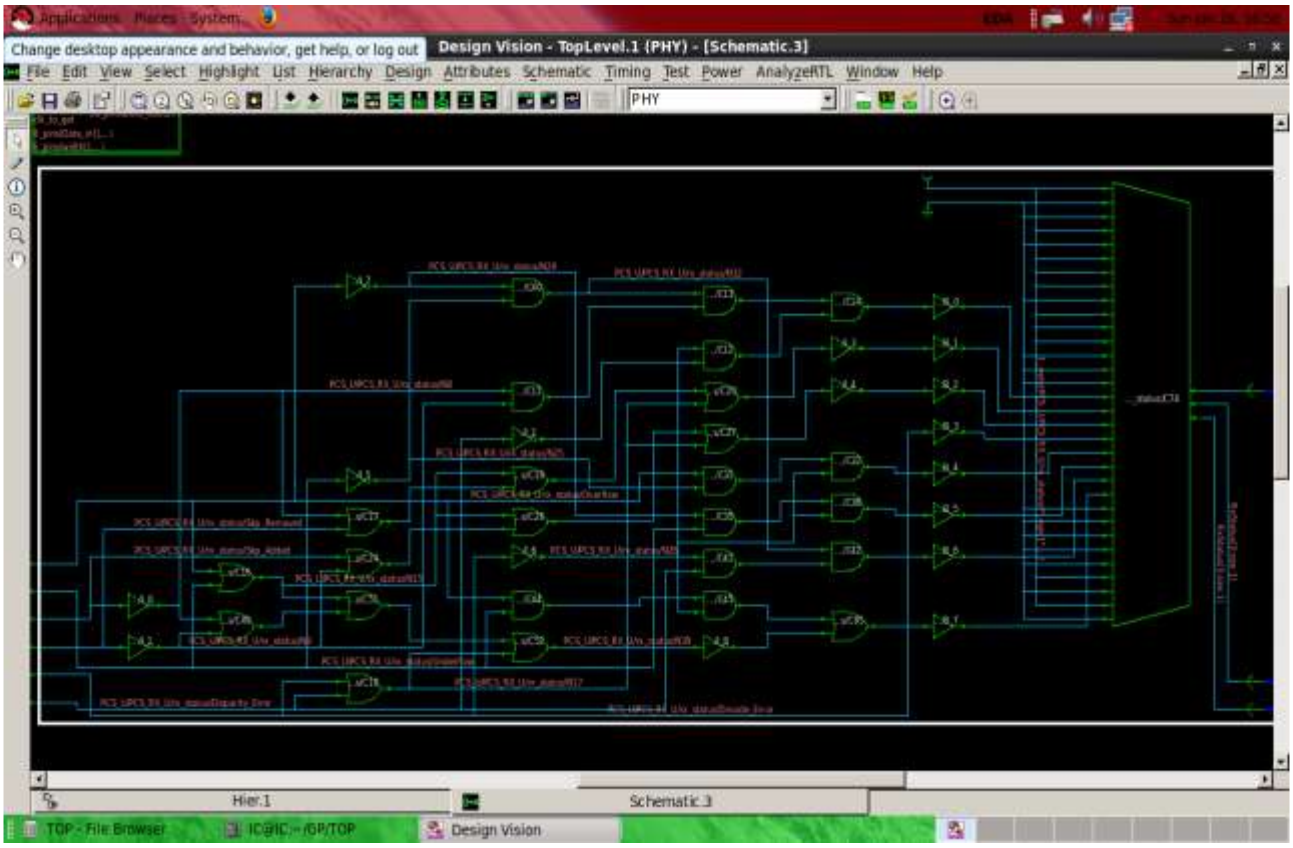

Design Vision - TopLevel.1 (PHY) - [Schematic.3]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.3
Design Vision

Figure 29 Reciever Status Synthesis

Comma Detection:

The module checks on every 10 bits of received data and raise a flag when a comma is detected to be able to read the data after it and store it in the elastic Buffer. This helps to align data as when a comma is received a flag is raised so the start and end of the 10 bits are known

Name	Direction	Width	Description
CLK	Input	1	Bit Rate Clk (5G)
Rst_n	Input	1	Active Low Reset
Detect_Comma	Input	10	Data collected serially
RxValid	Output	8	Data recovered from decoder
Comma_pulse	Output	1	Write pulse to write in buffer

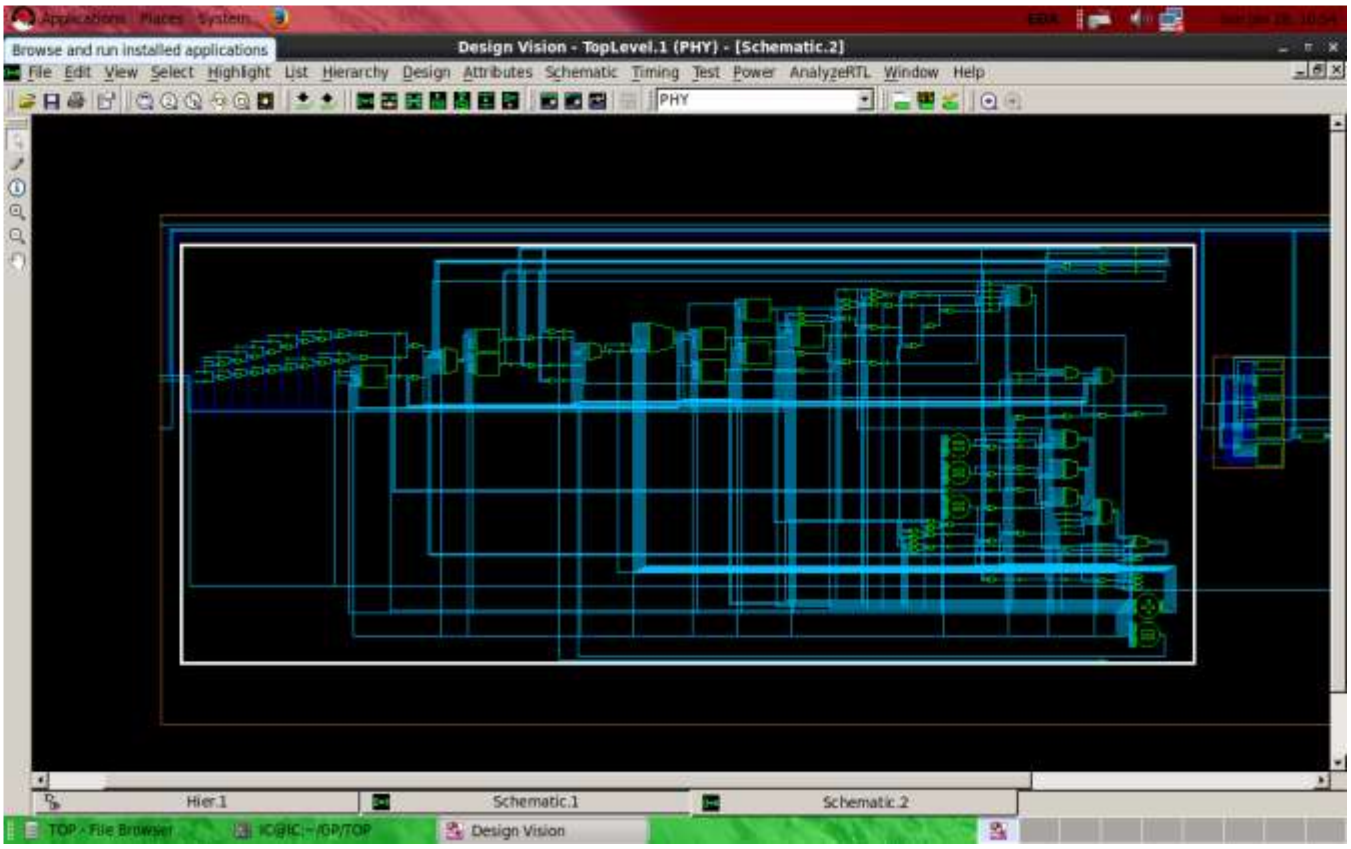

Applications Places System
Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.2]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.1
Schematic.2
TOP - File Browser
IC@IC:~/GP/TOP
Design Vision

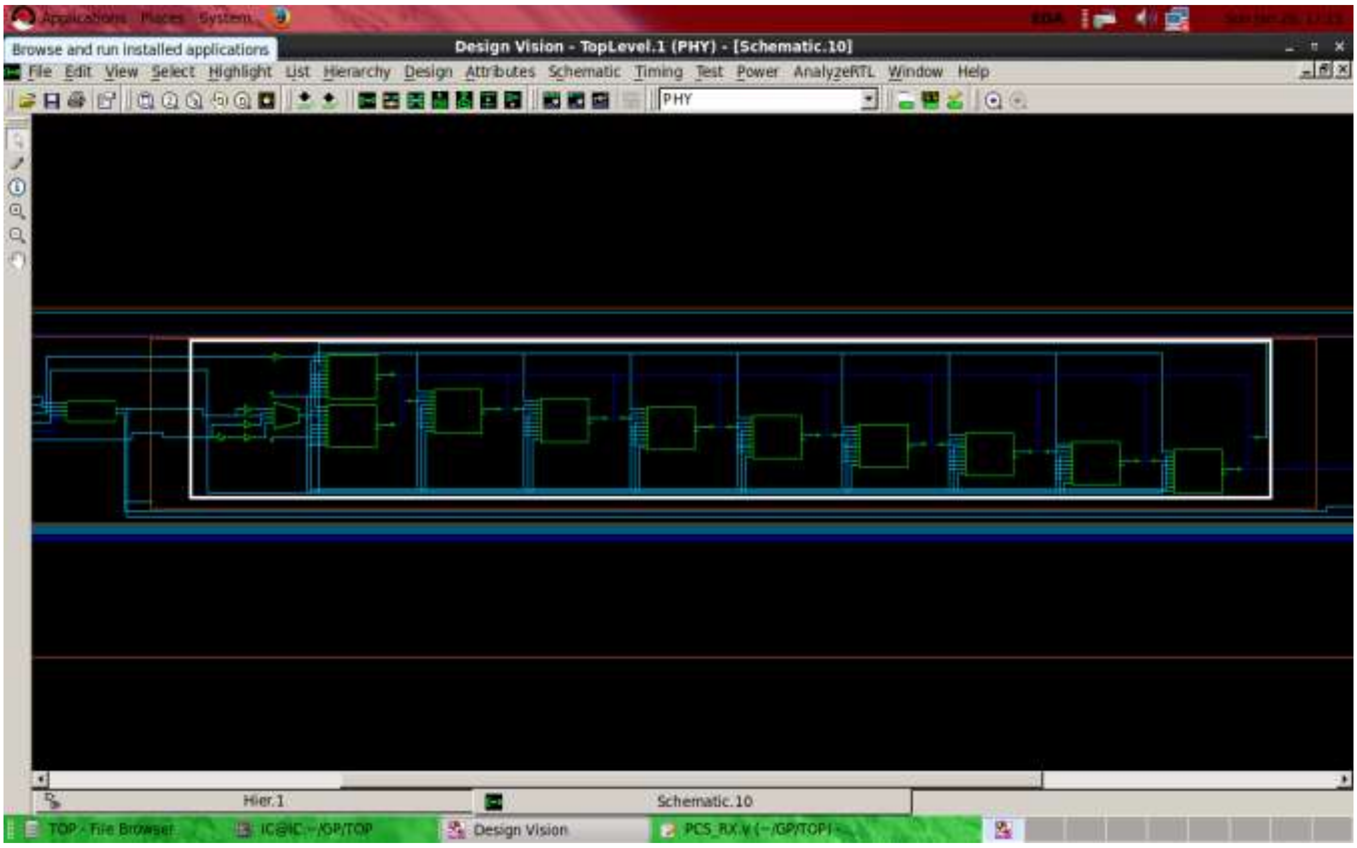
Figure 30 Comma detection Synthesis

PMA RX

Serial To parallel

After receiving the serial data from the channel it is transferred back to parallel to be decoded and received. Each 10 bits of data is collected into one parallel data

Name	Direction	Width	Description
Recovered_Bit_Clk	Input	1	Recovered CLK
Rst_n	Input	1	Active Low Reset
Ser_in	Input	1	Recovered serial bit from CDR
RXPolarity	Input	1	Indication to toggle serial bit or not
Data_to_Decoder	Output	10	Data recovered from decoder


Applications Places System
Browse and run installed applications
Design Vision - TopLevel.1 (PHY) - [Schematic.10]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.10
TOP - File Browser
IC@IC:~/GP/TOP
Design Vision
PCS_RX.v (~/GP/TOP) -

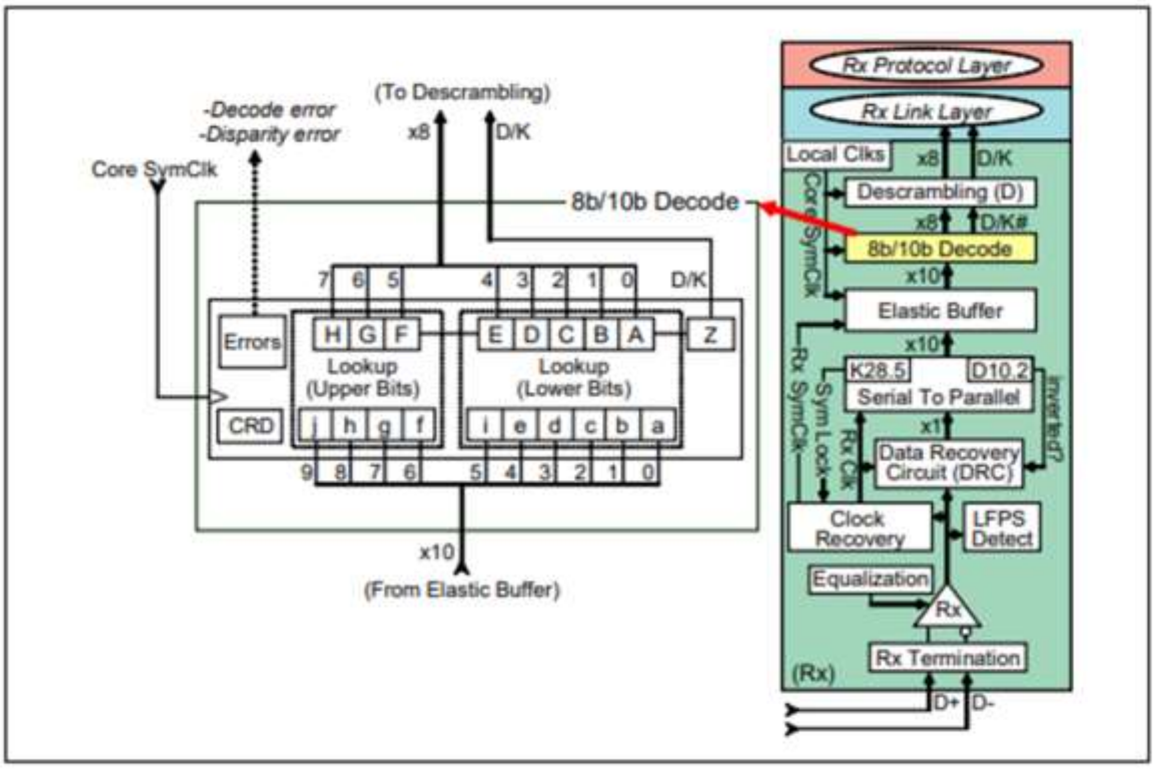
Decoder

After receiving the data as 10bits the decoder functionality is to map these bits back to the 8bits which is the original data. In addition to that it also define if this data is actual data or command by comparing where it got the equivalent 8bits.

An additional functionality is that it detects the errors which can be either decode error or disparity error.

Decode error is when the data is not found neither as positive encoding nor as positive encoding. On the other hand the disparity error occurs when the data is supposed to be positive encoding while the received was negative encoding or vice versa

Name	Direction	Width	Description
CLK	Input	1	WordCLK (125 , 250 , 500) MHz
Rst_n	Input	1	Active Low Reset
Data_in	Input	10	Encoded Data
Data_out	Output	8	Data recovered from decoder
DecodeError	Output	1	Error in Decoding
DisparityError	Output	1	Error in Data Disparity
RxDataK	Output	10	Indication whether it's actual data or command

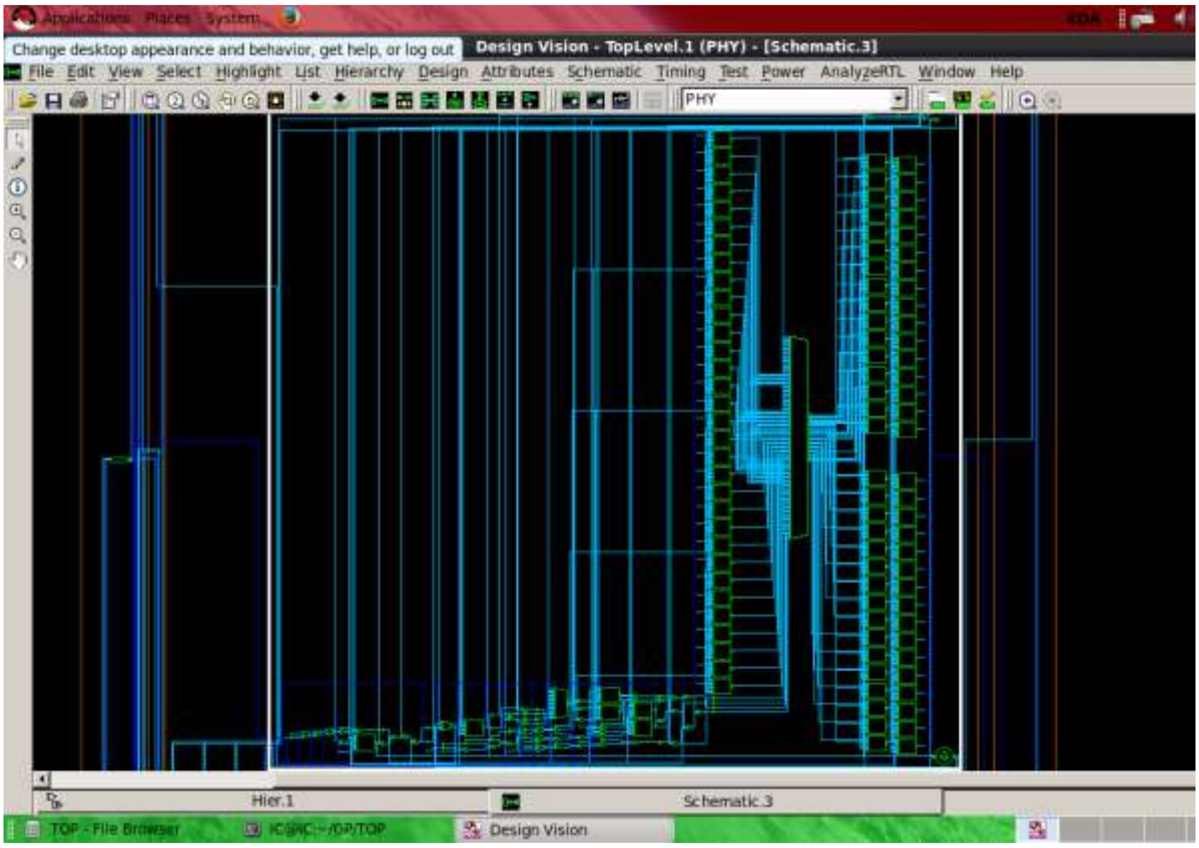


Gasket Rx:

In the receiver after receiving the blocks of 10bits and decoding them to 8bits. The gasket is responsible for collecting the 8-bit blocks according to the width

So the output become 8, 16 or 32 bit

Name	Direction	Width	Description
Word_Clk	Input	1	WordCLK (125 , 250 , 500) MHz
Rst_n	Input	1	Active Low Reset
PCLK	Input		Parallel Data CLK
Width	Input	6	Width of data (8,16,32)
Data_in	Input	8	Word Data
Data_out	Output	32	Data sent to MAC


Applications Places System
Change desktop appearance and behavior, get help, or log out
Design Vision - TopLevel.1 (PHY) - [Schematic.3]
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
PHY
Hier.1
Schematic.3
TOP - File Browser
ICS/IC:~/OP/TOP
Design Vision

CDR

In the context of USB (Universal Serial Bus) and data communication, "CDR" can stand for "Clock Data Recovery." Clock Data Recovery is a process that helps in the extraction of a clock signal from a data stream. It is particularly important in high-speed communication systems to ensure proper synchronization between the sender and receiver.

USB, especially in its SuperSpeed versions (USB 3.0 and later), uses complex signaling techniques to achieve high data transfer rates. These techniques involve the transmission of data along with a clock signal that indicates when the bits of data should be sampled. Clock Data Recovery becomes crucial in scenarios where the clock signal needs to be extracted accurately from the incoming data stream.

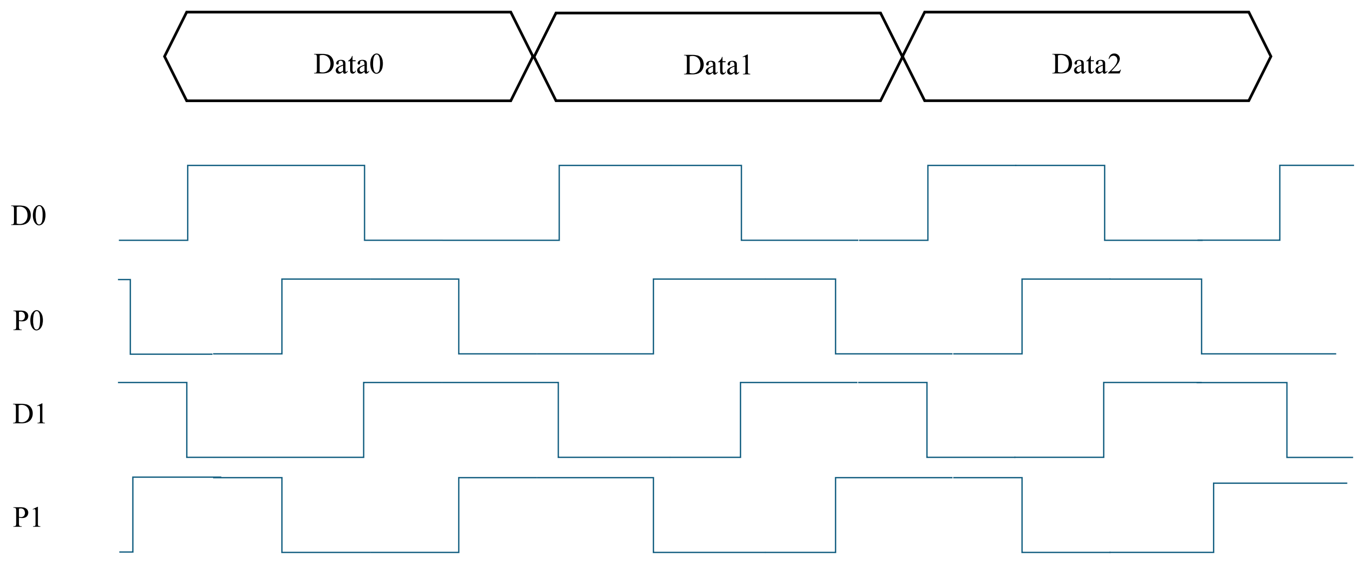
In USB SuperSpeed, the clock recovery process helps the receiver to synchronize with the incoming data stream, ensuring proper sampling and interpretation of the transmitted bits. This is important for maintaining signal integrity and reliable data communication.

CDR circuits are often implemented in the receiver circuitry of USB devices to extract and recover the clock signal from the incoming data. The recovered clock is then used to sample the data accurately, reconstructing the original information sent by the transmitter.

In summary, Clock Data Recovery (CDR) in USB involves the extraction of a clock signal from the incoming data stream, and it plays a critical role in ensuring reliable and high-speed data communication in USB systems.

The CDR designed is based on Bang Bang CDR. It operates on a simple principle of comparing the sampled data with two clock signals: the primary clock and a version shifted by 90 degrees, along with their inverses. The "bang-bang" terminology arises from the binary nature of the system – it makes binary decisions about clock adjustment. The system samples incoming data using the clock pair and their inverses. The sampled data is compared with the 90-degree shifted clock. The alignment with the rising or falling edge indicates whether the clock is early or late.

The Bang-Bang CDR's being simple in concept and implementation makes it an attractive choice In addition to that it provides robust clock recovery, exhibiting resilience to noise and variations in the incoming signal



In Bang Bang CDR there is 4clocks which sample the data. The comparison between the value given between P0-D0 or P0-D1 defines if the clock is early or late to be adjusted. If D0 =P0 then the data is early. On the other hand if P0=D1 then it late. For the CDR to work correctly there should be DC Balance and maximum run length as having a big number of consecutives ones or zeros will lead to the CDR not being able to detect if it early or late

The Clock and Data Recovery (CDR) mechanism with Bang Bang CDR is designed to ensure accurate synchronization of clocks and data in communication systems. The system utilizes four clocks for data sampling and relies on comparisons between P0-D0 and P0-D1 values to determine whether the clock is early or late, thereby facilitating necessary adjustments.

- Early Detection: If D0 equals P0, the system interprets the data as early.
- Late Detection: If P0 equals D1, the system identifies the data as late.

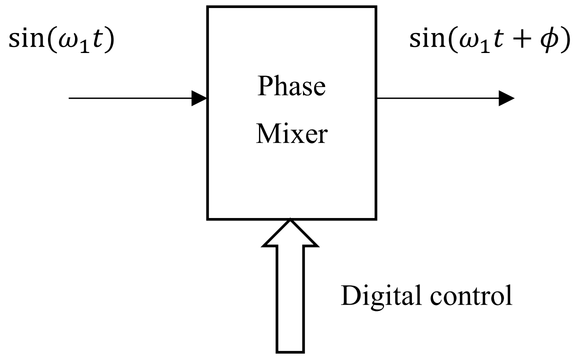
Maintaining DC balance is crucial for the proper functioning of the CDR. DC balance ensures an equilibrium between the number of ones and zeros in the data stream. An imbalance in DC levels can impact the CDR's ability to accurately detect whether the clock is early or late, potentially leading to synchronization issues.

The system takes into account the concept of maximum run length, which refers to the maximum number of consecutive ones or zeros in the data stream. Monitoring and limiting the run length are essential to prevent challenges in accurate clock and data recovery. Excessive consecutive ones or zeros may hinder the CDR's ability to discern the timing relationship between the clock and data.

Phase Mixer

This module is part of the CDR it is essential to change the phase and frequency of the clock to synchronize it with the received data to be able to sample the data correctly without losing any bits

The input to the phase mixer is the clock and the digital control, which defines how the output will change. If the digital control keeps changing the frequency will change as the ϕ will be function of frequency

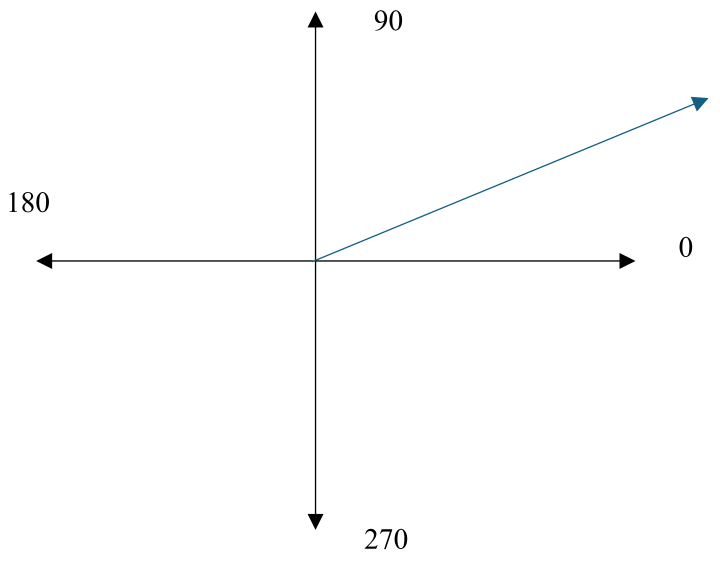


If $\phi(t) = k \rightarrow \sin(\omega t + k) \rightarrow change\ in\ phase$

$\phi(t) = kt \rightarrow \sin(\omega t + kt) \rightarrow change\ in\ frequency$

$\phi(t) = kt^2 \rightarrow \sin(\omega t + kt) \rightarrow spreading$

The Clock and Data Recovery (CDR) Phase Mixer plays a pivotal role in digital communication systems. It utilizes a digital code structure to select two phases from a predefined set, employing the first n bits for this purpose. The remaining bits finely control interpolation between the selected phases. In an example scenario with 4 phases (0, 90, 180, 270 degrees), 2 bits choose two phases, while 8 bits govern precise interpolation. This mechanism, often used in high-speed interfaces, optimizes recovered clock signals, ensuring reliable data recovery by adapting to variations in incoming clocks. Its flexibility and high-resolution interpolation contribute to the system's overall reliability and data integrity.



Simulink model for Phase Mixer

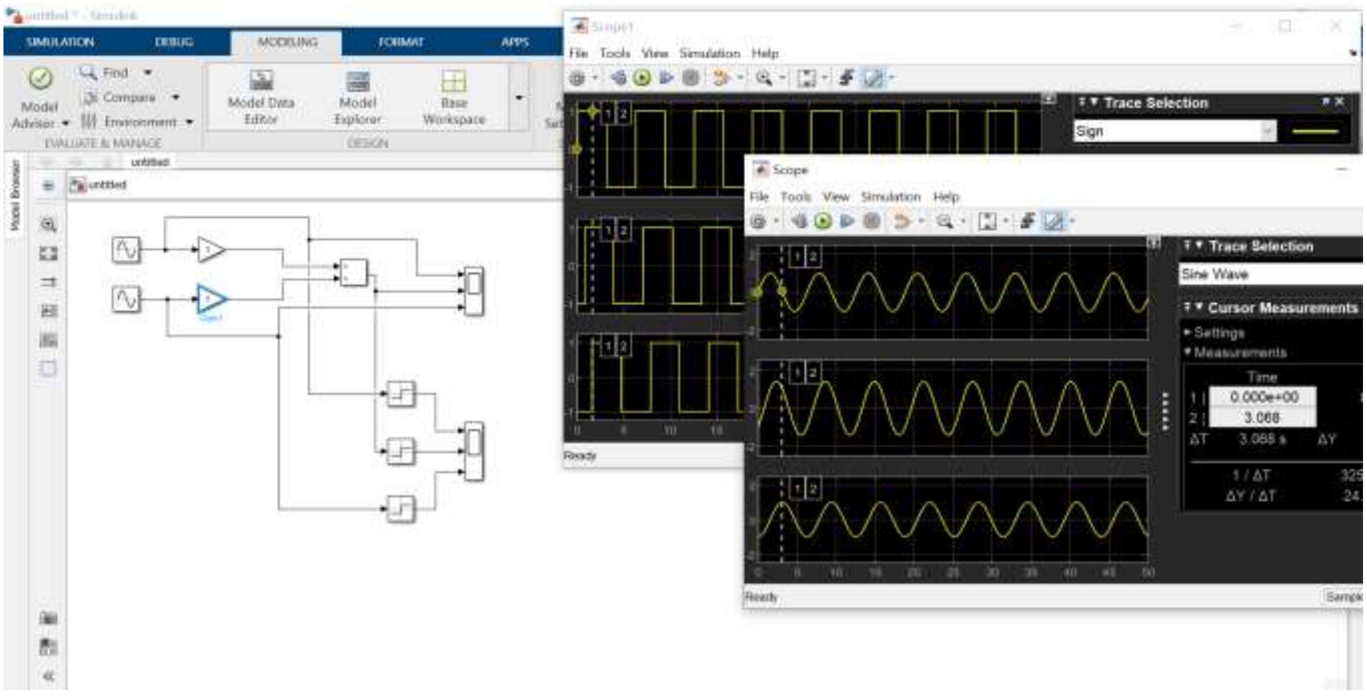
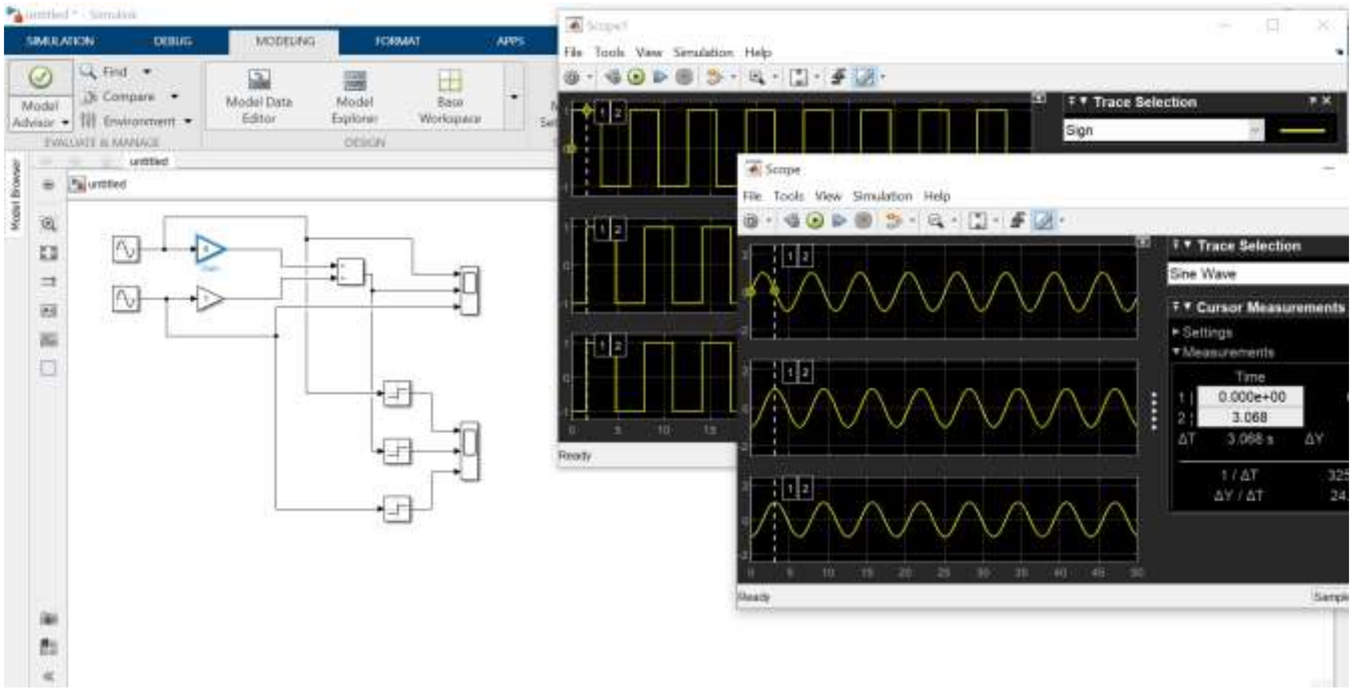
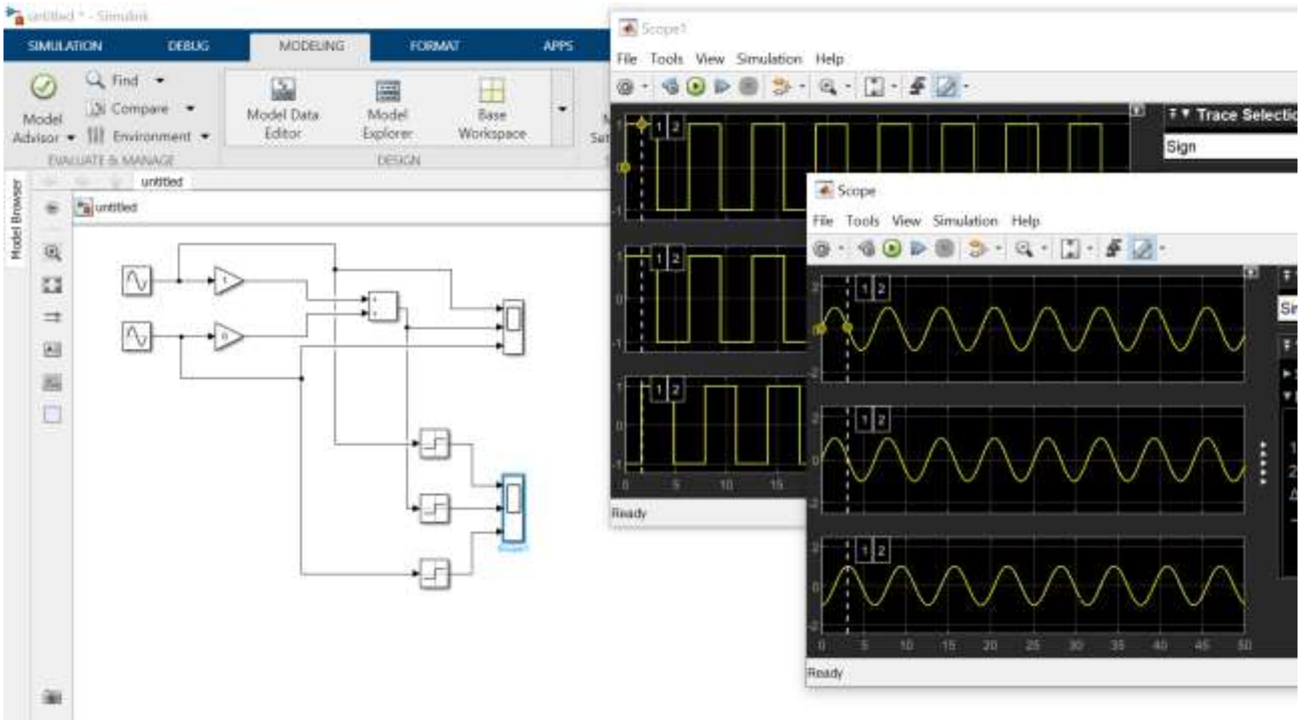
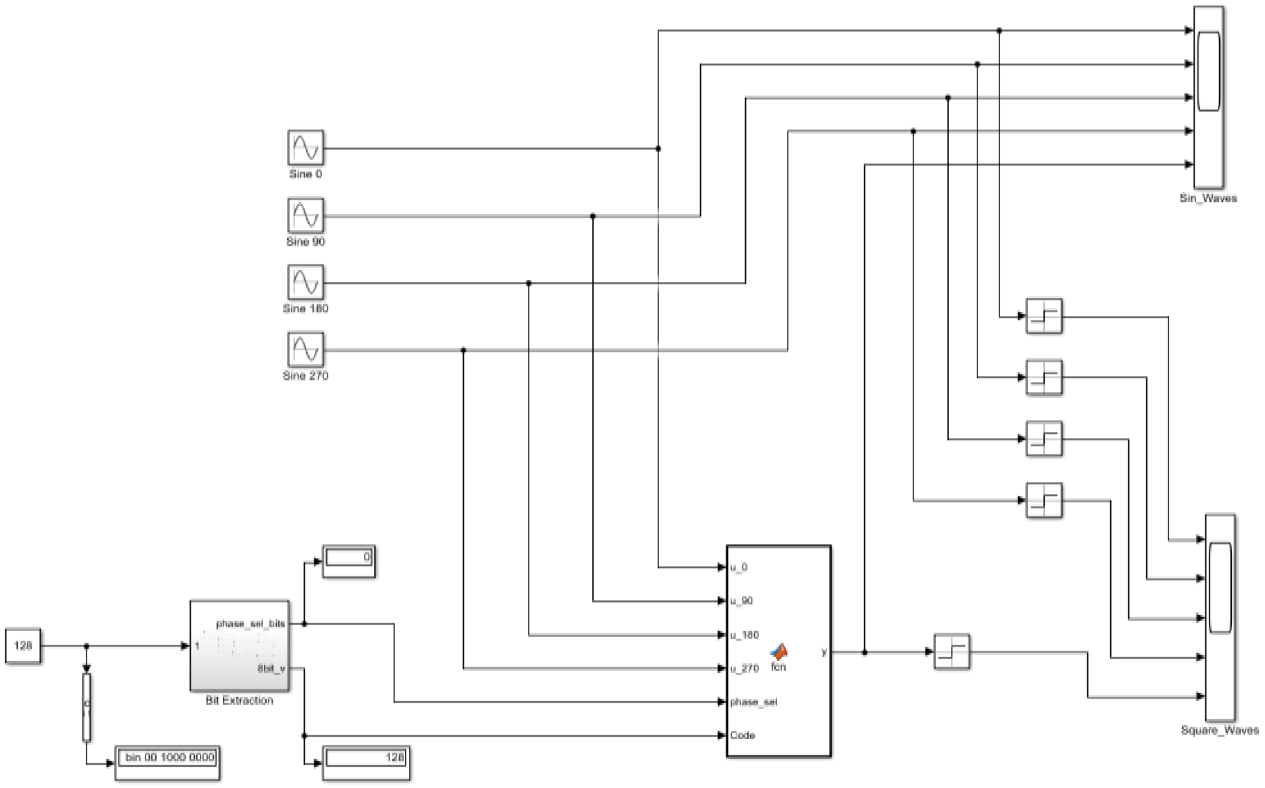


Figure 31 Phase Mixer models

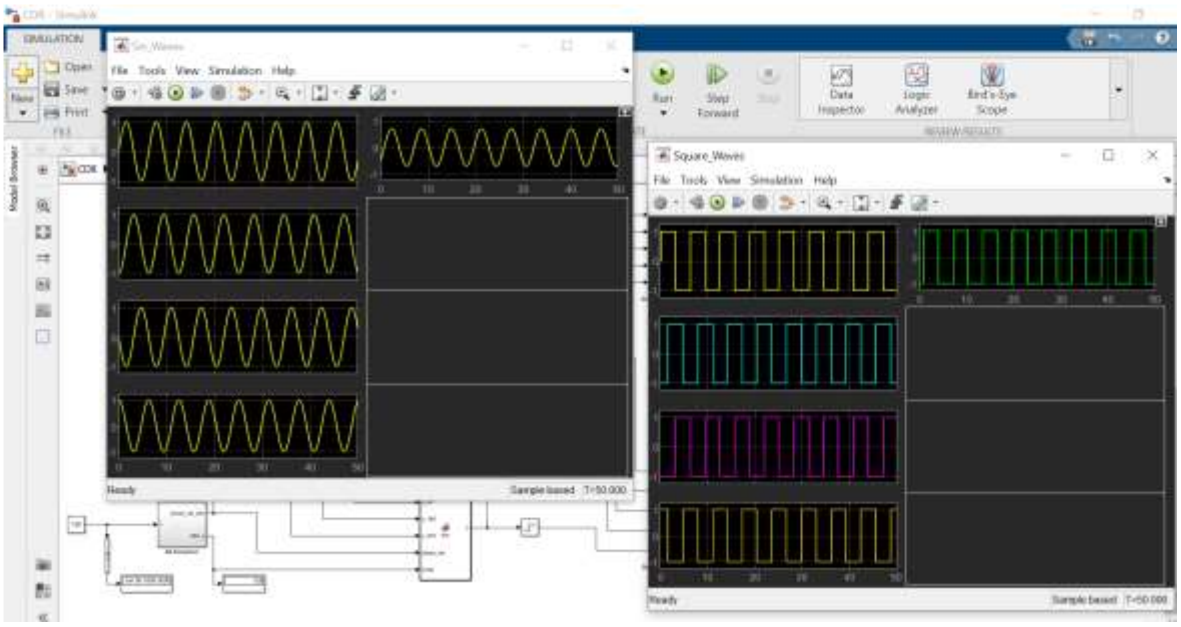


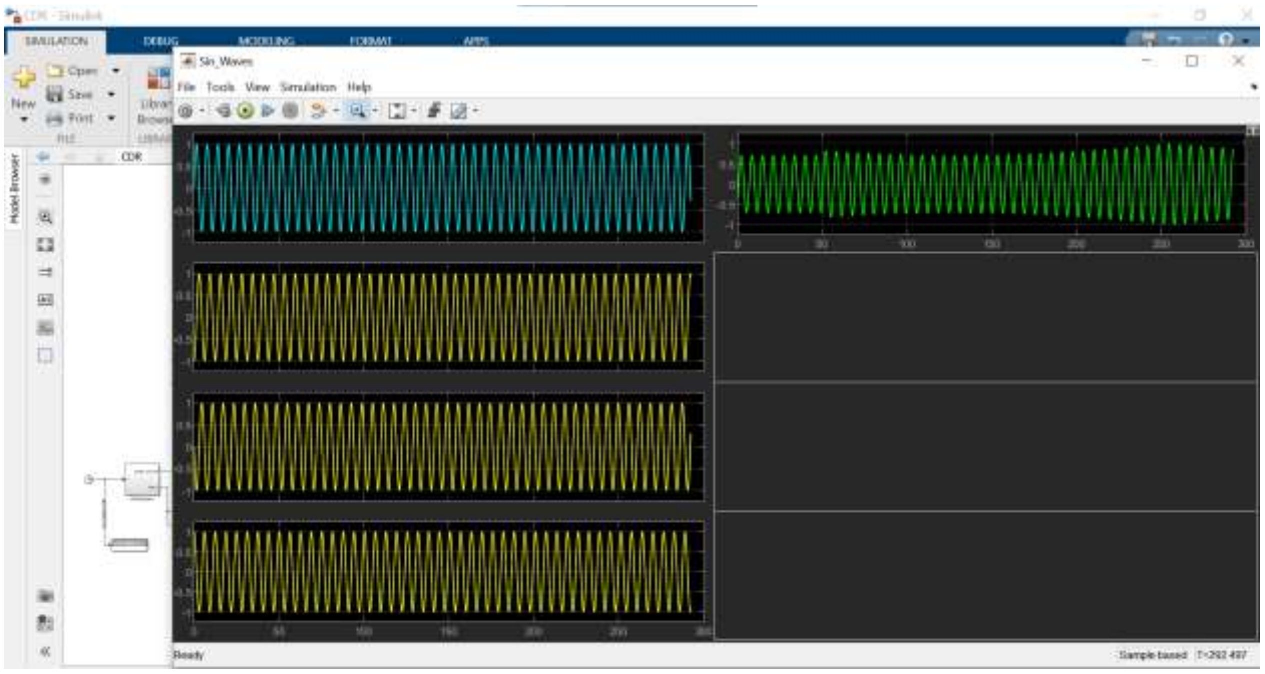


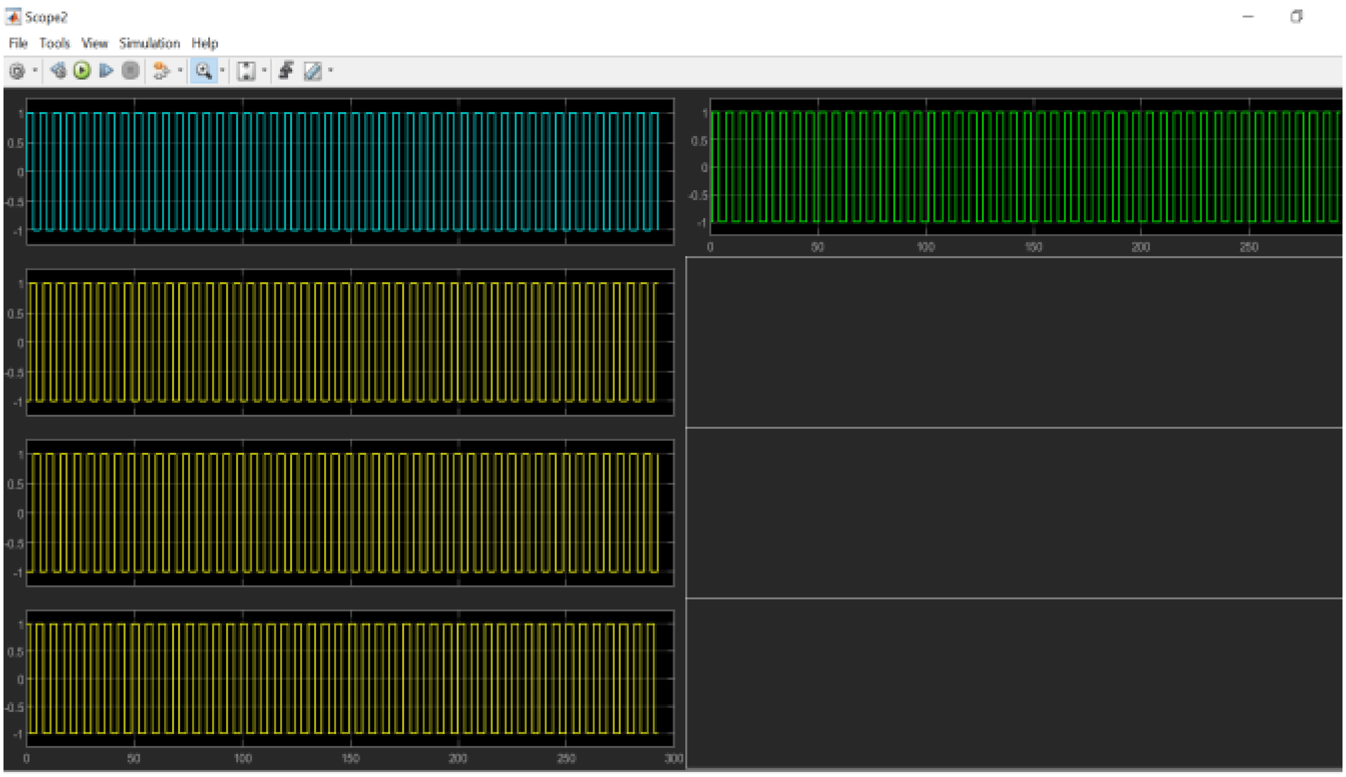
untitled * - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Model Advisor
Find
Compare
Environment
EVALUATE & MANAGE
Model Data Editor
Model Explorer
Base Workspace
DESIGN
Model Browser
untitled
Scope1
File Tools View Simulation Help
Trace Selection
Sign
Scope
Ready

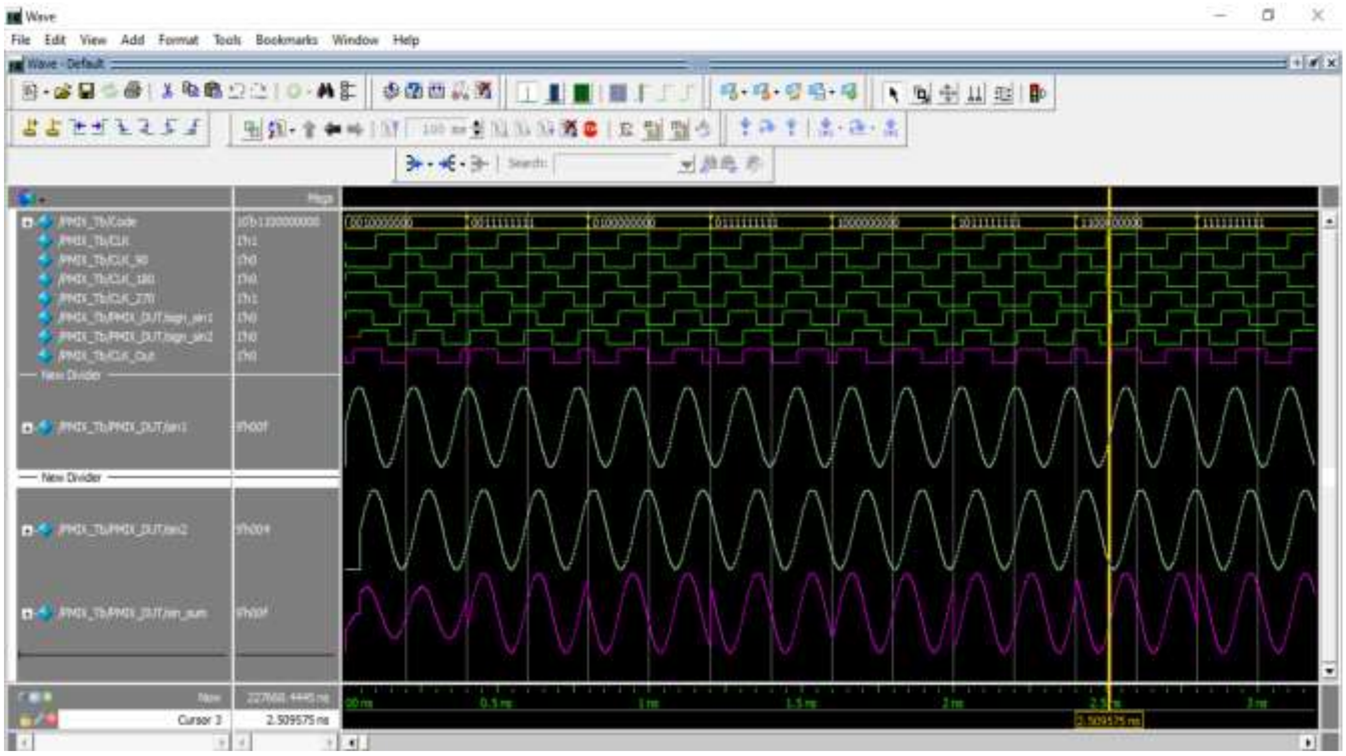


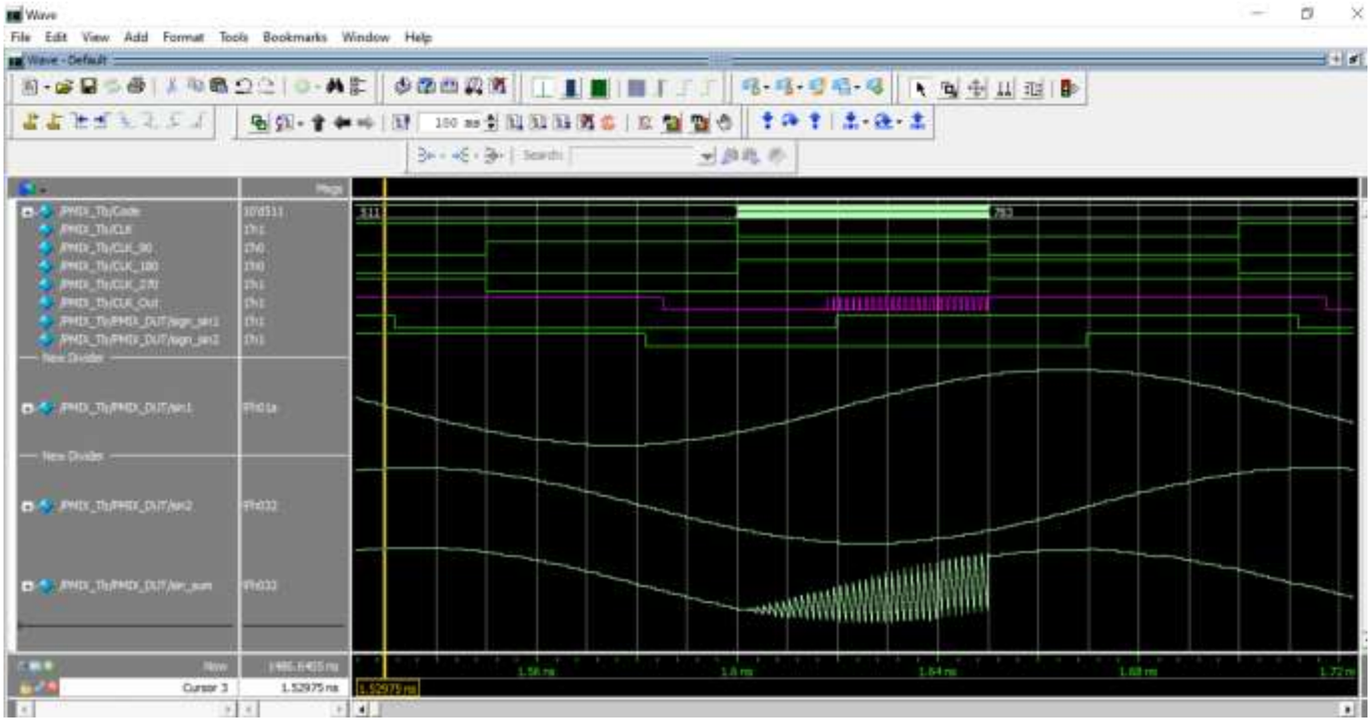
Sine 0
Sine 90
Sine 180
Sine 270
Sin_Waves
Square_Waves
128
phase_sel_bits
8bit_v
Bit Extraction
0
bin 00 1000 0000
128
u_0
u_90
u_180
u_270
phase_sel
Code
fcn
y

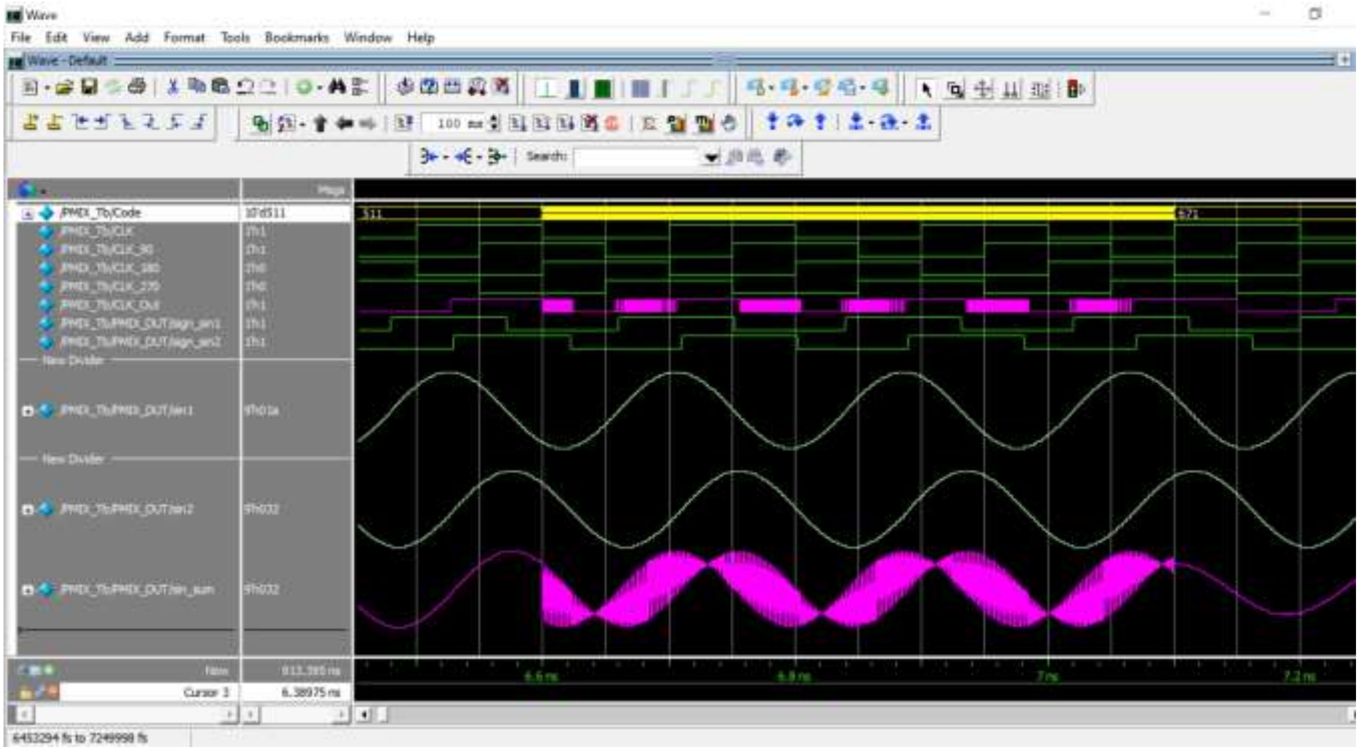

Sin_Waves
File Tools View Simulation Help
Square_Waves
Run
Step Forward
Stop
Data Inspector
Logic Analyzer
Bird's-Eye Scope
Ready
Sample based T=50.000

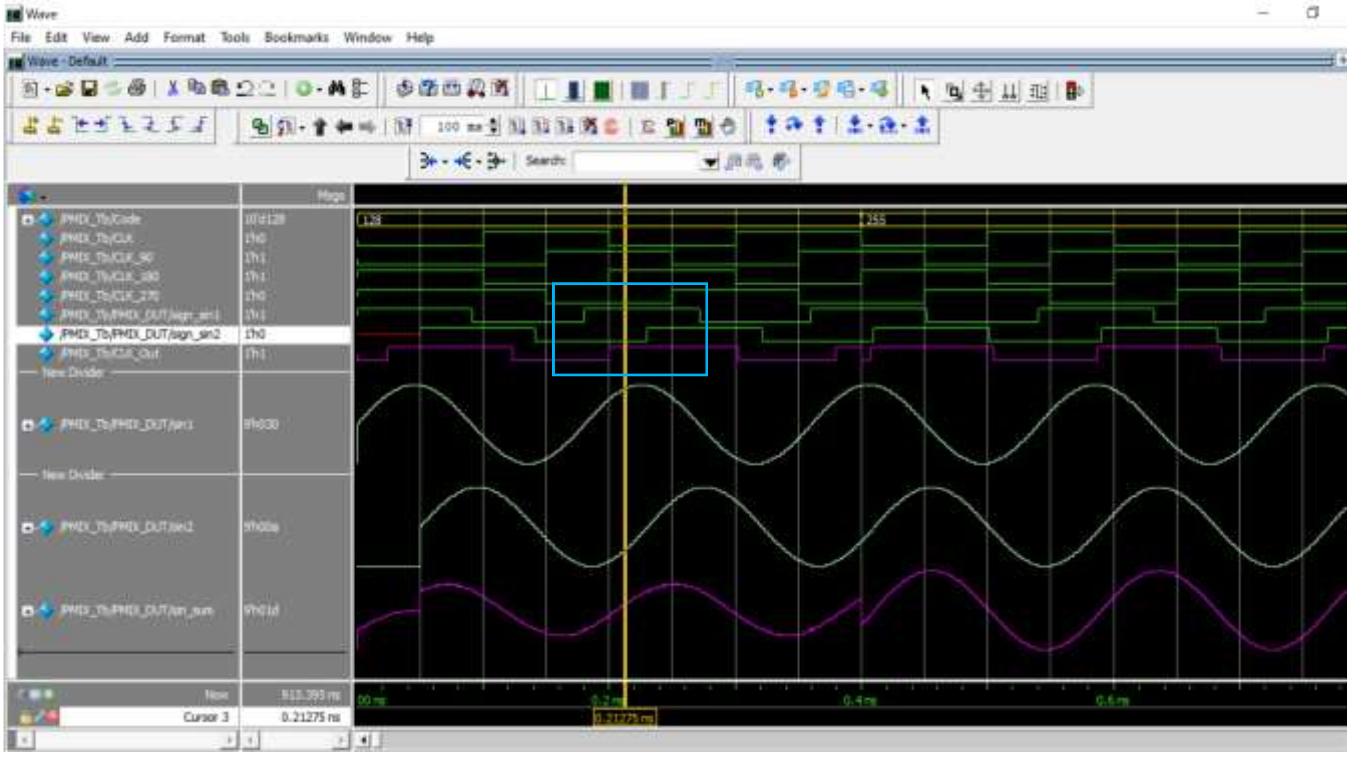

SIMULATION
DEBUG
MODELING
FORMAT
APPS
Sin_Waves
File Tools View Simulation Help
New
Open
Save
Print
Ready
Sample based


Scope2
File Tools View Simulation Help
1
0.5
0
-0.5
-1
0 50 100 150 200 250 300


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Msgs
/PMIX_Tb/Code
/PMIX_Tb/CLK
/PMIX_Tb/CLK_90
/PMIX_Tb/CLK_180
/PMIX_Tb/CLK_270
/PMIX_Tb/PMIX_DUT/sign_sin1
/PMIX_Tb/PMIX_DUT/sign_sin2
/PMIX_Tb/CLK_Out
New Divider
/PMIX_Tb/PMIX_DUT/sin1
New Divider
/PMIX_Tb/PMIX_DUT/sin2
/PMIX_Tb/PMIX_DUT/sin_sum
Now
Cursor 3
2.509575 ns
0 ns
0.5 ns
1 ns
1.5 ns
2 ns
2.5 ns
3 ns


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Msgs
/PMDX_Tb/Code 10'd511
/PMDX_Tb/CLK 1'h1
/PMDX_Tb/CLK_90 1'h0
/PMDX_Tb/CLK_180 1'h0
/PMDX_Tb/CLK_270 1'h1
/PMDX_Tb/CLK_Out 1'h1
/PMDX_Tb/PMDX_DUT/sign_sin1 1'h1
/PMDX_Tb/PMDX_DUT/sign_sin2 1'h1
New Divider
/PMDX_Tb/PMDX_DUT/sin1 9'h01a
New Divider
/PMDX_Tb/PMDX_DUT/sin2 9'h032
/PMDX_Tb/PMDX_DUT/sin_sum 9'h032
Now 1486.6455 ns
Cursor 3 1.52975 ns
511
783
1.56 ns
1.6 ns
1.64 ns
1.68 ns
1.72 ns
1.52975 ns


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Msgs
/PMDX_Tb/Code 10'd511
/PMDX_Tb/CLK 1'h1
/PMDX_Tb/CLK_90 1'h1
/PMDX_Tb/CLK_180 1'h0
/PMDX_Tb/CLK_270 1'h0
/PMDX_Tb/CLK_Out 1'h1
/PMDX_Tb/PMDX_DUT/sign_sin1 1'h1
/PMDX_Tb/PMDX_DUT/sign_sin2 1'h1
New Divider
/PMDX_Tb/PMDX_DUT/sin1 9'h01a
New Divider
/PMDX_Tb/PMDX_DUT/sin2 9'h032
/PMDX_Tb/PMDX_DUT/sin_sum 9'h032
Now 813.395 ns
Cursor 3 6.38975 ns
511
671
6.6 ns
6.8 ns
7 ns
7.2 ns
6453294 fs to 7249998 fs



	Msgs
/PMIX_Tb/Code	10'd128
/PMIX_Tb/CLK	1'h0
/PMIX_Tb/CLK_90	1'h1
/PMIX_Tb/CLK_180	1'h1
/PMIX_Tb/CLK_270	1'h0
/PMIX_Tb/PMIX_DUT/sign_sin1	1'h1
/PMIX_Tb/PMIX_DUT/sign_sin2	1'h0
/PMIX_Tb/CLK_Out	1'h1
/PMIX_Tb/PMIX_DUT/sin1	9'h030
/PMIX_Tb/PMIX_DUT/sin2	9'h00a
/PMIX_Tb/PMIX_DUT/sin_sum	9'h01d
Now	813.393 ns
Cursor 3	0.21275 ns

PHY Verification

This section verifies the functionality and performance of the PHY the top module design, which implements the USB protocol for data communication. The verification objectives are to ensure that the PHY design conforms to the USB protocol specifications, and that it meets the required functionality ,throughput, latency, targets. The verification plan outlines the verification strategy, methodology, environment, scenarios, and criteria for the PHY design. The verification environment is based on the UVM framework, which provides a standardized and reusable way of creating and executing testbenches for digital designs and systems-on-chip (SoCs).

UVM is a verification methodology that uses a SystemVerilog-based, object-oriented approach to create modular, configurable, and scalable testbench components that can be easily integrated into the verification process. UVM also provides guidelines and best practices for developing testbenches, running simulations, and analyzing results. UVM consists of several key components, such as drivers, monitors, scoreboards, agents, sequences, and environments, that perform different functions and roles in the verification process. UVM also supports transaction-level modeling (TLM), which enables the communication and synchronization between testbench components using abstract transactions rather than signals.

The testbench architecture for the PHY verification is shown in Figure. The testbench consists of two main components: the UVM test and the UVM environment. The UVM test is responsible for generating and controlling the test scenarios and stimuli for the DUT. The UVM environment is responsible for providing the testbench components and interfaces for the DUT. The testbench components include the following:

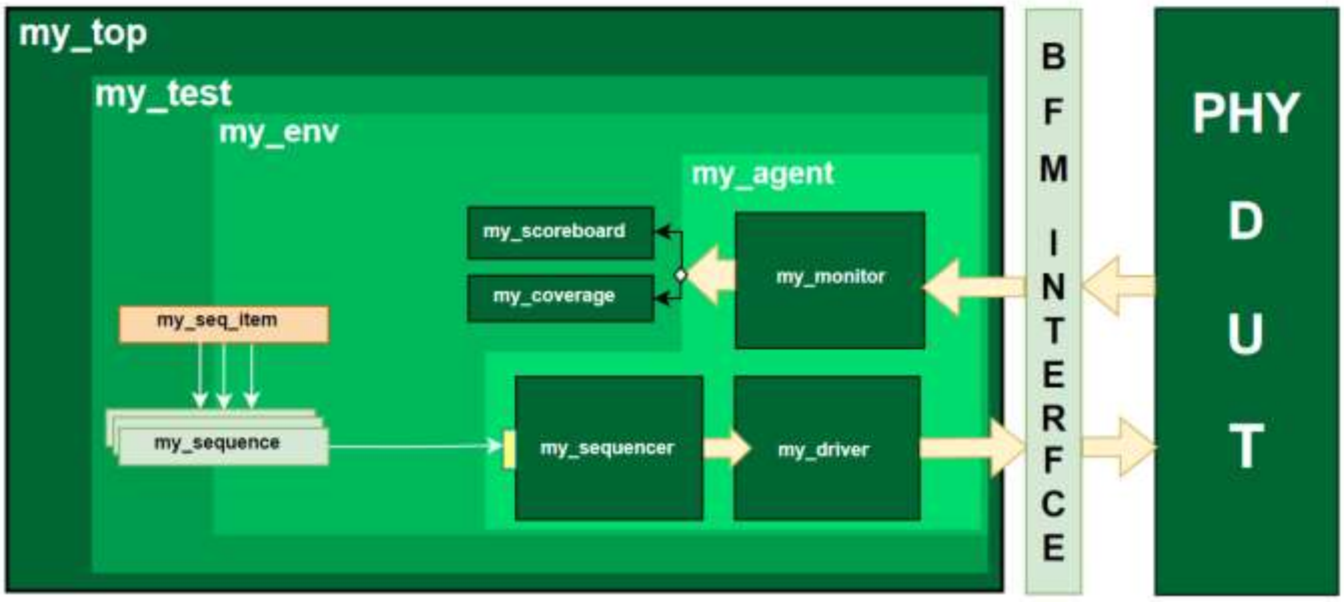


Figure 32 UVM Structure

My agent, which models the behavior and protocol of the PHY interface, and consists of a driver, a monitor, and a sequencer.

My driver, which is responsible for driving the signals from the sequencer to the Bus Functional Model wich passes the data to the PHY DUT.

My Monitor, which responsible for monitoring the output data from the DUT and sending it using TLM to Scoreboard and coverage components.

The scoreboard, which compares the expected and actual outputs of the DUT, and reports any mismatches or errors.

The coverage collector, which collects and analyzes the functional and code coverage of the DUT, and reports the coverage metrics and goals.

The testbench interfaces include the following:

The BFM interface, which connects the my_driver to the PHY port of the DUT, and provides the signals and transactions for the USB protocol.

The testbench components and interfaces are configured and connected using UVM configuration and TLM mechanisms, which allow the flexibility and reusability of the testbench architecture. The testbench components and interfaces interact with the DUT and each other using UVM sequences which are predefined or random sequences of transactions that drive and monitor the DUT behavior and performance through different phases organizes the time line of the verification process and that is one of most important features in UVM all phases are indicated in the figure:

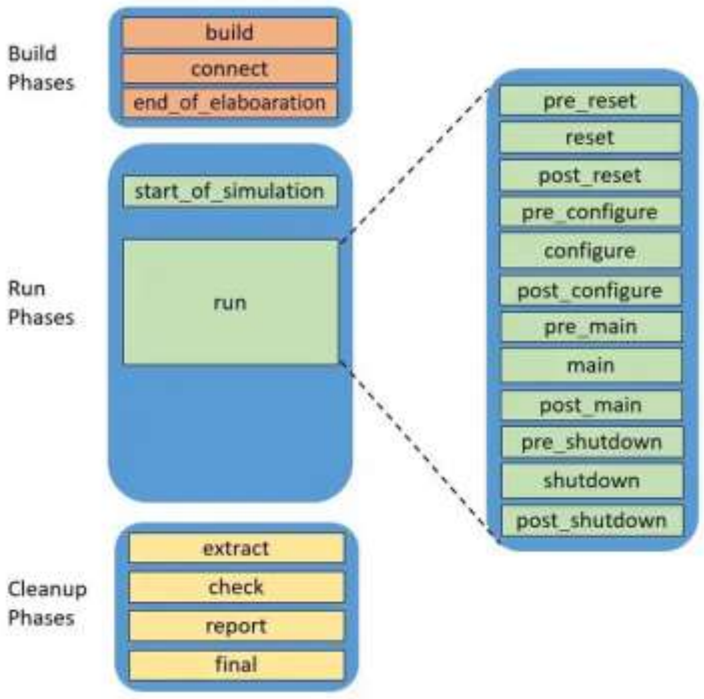


Figure 33 UVM Phases

Build phases: These phases are used to construct, configure, and connect the testbench components. They are executed at the start of the simulation, and they do not consume simulation time. The build phases are:

- build_phase: Used to create the instances of the testbench components and set their initial configuration.
- connect_phase: Used to connect the TLM ports and exports of the components using TLM connections or uvm_config_db.
- end_of_elaboration_phase: Used to make any final adjustments to the testbench structure, configuration, or connectivity before the simulation starts.

Run-time phases: These phases are used to generate, drive, monitor, and check the test stimuli and responses. They are executed in parallel with the run_phase, which is the main phase that consumes simulation time. The run-time phases are:

- start_of_simulation_phase: Used to display the testbench topology or configuration information, or to set any run-time configuration.
- run_phase: Used to generate and control the test scenarios and stimuli for the DUT, and to monitor and check the DUT behavior and performance.
- pre_reset_phase: Used to perform any activity that should occur before the reset, such as waiting for a power-good signal.
- reset_phase: Used to generate a reset and to put the DUT or interface into its default state.
- post_reset_phase: Used to perform any activity that should occur after the reset, such as configuring the DUT or interface.
- pre_configure_phase: Used to perform any activity that should occur before the configuration, such as setting the initial seed or randomization constraints.
- configure_phase: Used to configure the DUT or interface with the required parameters or registers.
- post_configure_phase: Used to perform any activity that should occur after the configuration, such as checking the configuration status or enabling the interrupts.
- pre_main_phase: Used to perform any activity that should occur before the main phase, such as setting the initial coverage or scoreboard settings.
- main_phase: Used to generate and drive the main test stimuli and responses, and to monitor and check the main DUT behavior and performance.
- post_main_phase: Used to perform any activity that should occur after the main phase, such as collecting the final coverage or scoreboard data.
- pre_shutdown_phase: Used to perform any activity that should occur before the shutdown, such as sending the end-of-test signal or flushing the queues.
- shutdown_phase: Used to shut down the DUT or interface gracefully, and to perform any cleanup or recovery actions.
- post_shutdown_phase: Used to perform any activity that should occur after the shutdown, such as checking the shutdown status or releasing the resources.

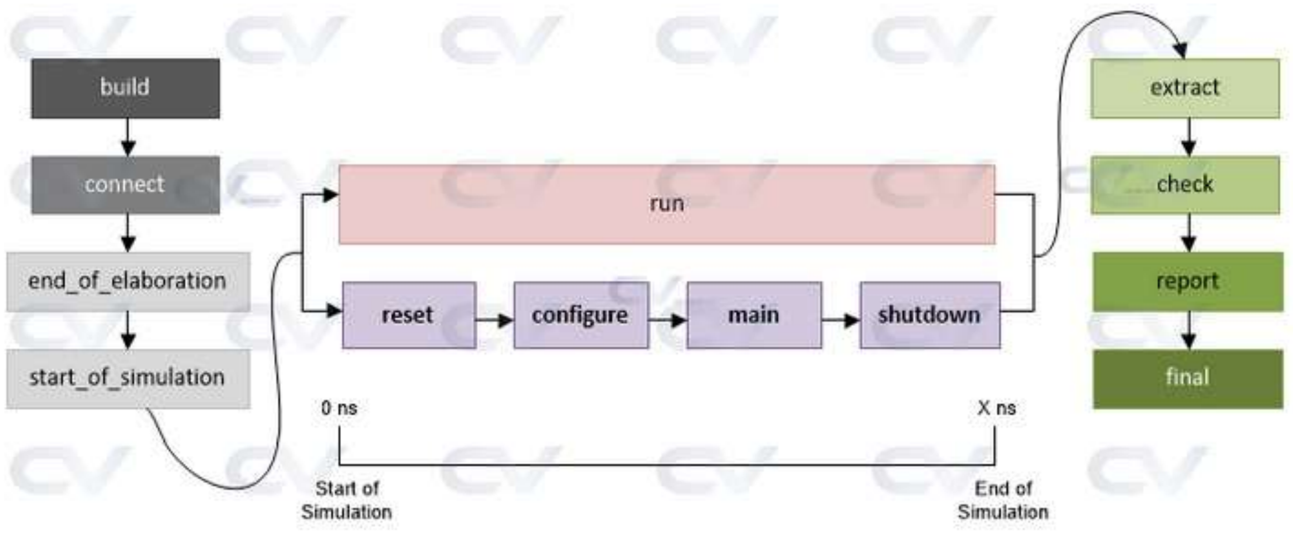
Clean-up phases: These phases are used to extract, check, and report the verification results. They are executed at the end of the simulation, and they do not consume simulation time. The clean-up phases are:

extract_phase: Used to retrieve and process the information from the scoreboards and functional coverage monitors, and to compute the expected data or coverage metrics.

check_phase: Used to check that the DUT behaved correctly and to identify any errors or mismatches that may have occurred during the simulation.

report_phase: Used to display or write the verification results, such as the test status, errors, warnings, messages, coverage, and metrics.

final_phase: Used to complete any other outstanding actions that the testbench has not already completed, such as closing the files or terminating the processes.



Test Strategy

The verification Features for the PHY design verification are summarized in the Table. The verification results and waveforms show that the PHY design passes all the functional and performance tests, and achieves the required coverage and metrics. The verification results are verified and validated using UVM reports, which provide the detailed information and statistics of the verification process, such as the test name, status, duration, errors, warnings, messages, coverage, and metrics.

Reset_n	DataBus Width	MAC_TX _Data	MAC_TX _DataK	MAC_Data _En	RxPolarity	Test Feature
0	8	x	x	1	0	Resetting the Design
0	8	x	x	0	1	
0	16	x	x	1	0	
0	16	x	x	0	0	
0	32	x	x	1	0	
0	32	x	x	0	0	
1	8	32'd0000BC	1	1	0	Collecting Data S to P with COMMA Detection with DATA WIDTH = 8
1	8	Random 32bit data	0	1	0	
1	8	32'd0000BC	1	1	0	
1	8	Random 32bit data	0	1	0	
1	8	32'dxxxxBC	1	1	0	
1	8	Random 32bit data	0	1	0	
1	8	32'dxxxxBC	1	1	0	
1	8	Random 32bit data	0	1	0	
1	16	32'd0000BC BC	1	1	0	Collecting Data S to P with COMMA Detection with DATA WIDTH = 16
1	16	Random 32bit data	0	1	0	
1	16	32'd0000BC BC	1	1	0	

1	16	Random 32bit data	0	1	0	
1	16	32’dxxxxBCBC	1	1	0	
1	16	Random 32bit data	0	1	0	
1	16	32’dxxxxBCBC	1	1	0	
1	32	32’dBCBCBCBC	1	1	0	Collecting Data S to P with COMMA Detection with DATA WIDTH = 32
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	32’dBCBCBCBC	1	1	0	Elastic Buffer Underflow & RX_Status = 3’b110
1	32	Random 32bit data	0	1	0	
1	16	32’dBCBCBCBC	1	1	0	
1	16	Random 32bit data	0	1	0	
1	8	32’dBCBCBCBC	1	1	0	

1	8	Random 32bit data	0	1	0	
1	32	32’dBCBCB CBC	1	1	0	Elastic Buffer Underflow & RX_Status = 3’b110
1	32	Random 32bit data	0	1	0	
1	16	32’dBCBCB CBC	1	1	0	
1	16	Random 32bit data	0	1	0	
1	8	32’dBCBCB CBC	1	1	0	
1	8	32’dBCBCB CBC	1	1	0	
1	32	32’dBCBCB CBC	**1**	**1**	**0**	**Elastic Buffer SKP ADDED & RX_Status = 3’b001**
1	32	32’dBCBCB CBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCB CBC	1	1	0	
1	32	32’dBCBCB CBC	1	1	0	

1	32	32’dBCBCBCBC	1	1	0	Elastic Buffer SKP ADDED & RX_Status = 3’b001
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	32’dBCBCBCBC	1	1	0	Elastic Buffer SKP Removed & RX_Status = 3’b010
1	32	Random 32bit data	0	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	Decoder Decode Error & RX_Status = 3’b100
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
1	32	32’dBCBCBCBC	1	1	0	Decoder Disparity Error

1	32	Random 32bit data	0	1	0	& RX_Status = 3’b111
1	32	32’dBCBCBCBC	1	1	0	
1	32	Random 32bit data	0	1	0	
1	32	32’dBCBCBCBC	1	1	0	
Random Patterns Generating With Different Seeds						Testing All Main Features Power and Reset Test & Stress Testing
Coverage Driven Verification Using Functional Coverage and Assertions						Testing All Events & generated Clocks Period

Waveform

- <u>Generated Clocks periods based on the Data width Used</u>

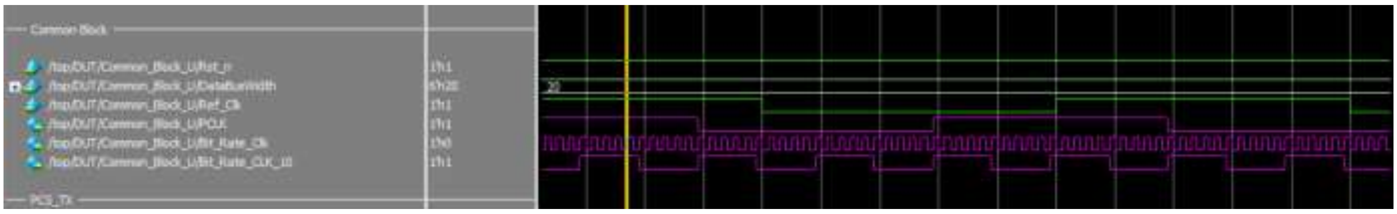

Common Block
/top/DUT/Common_Block_U/Rst_n
/top/DUT/Common_Block_U/DataBusWidth
/top/DUT/Common_Block_U/Ref_Clk
/top/DUT/Common_Block_U/PCLK
/top/DUT/Common_Block_U/Bit_Rate_Clk
/top/DUT/Common_Block_U/Bit_Rate_CLK_10
1'h1
6'h20
1'h1
1'h1
1'h0
1'h1
20
PCS_TX

Figure 34 Width = 32 bit clock generated

- For the (Width = 32) bit the Clocks generated from the Common Block and extracted from (Ref_CLk = 100MHz):

- *PCLK running at 125MHz.*
- *Symbol_Clk running at 500MHz.*
- *Bit_Rate_Clk running at 5GHz.*

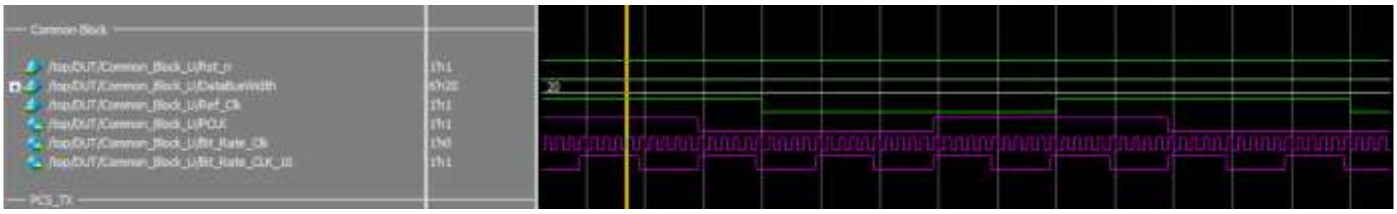

Common Block
/top/DUT/Common_Block_U/Rst_n
/top/DUT/Common_Block_U/DataBusWidth
/top/DUT/Common_Block_U/Ref_Clk
/top/DUT/Common_Block_U/PCLK
/top/DUT/Common_Block_U/Bit_Rate_Clk
/top/DUT/Common_Block_U/Bit_Rate_CLK_10
1'h1
6'h20
1'h1
1'h1
1'h0
1'h1
20
PCS_TX

Figure 35 Width = 16 bit clock generated

- For the (Width = 16) bit the Clocks generated from the Common Block and extracted from

(Ref_CLk = 100MHz):

- *PCLK running at 250MHz.*
- *Symbol_Clk running at 500MHz.*
- *Bit_Rate_Clk running at 5GHz.*

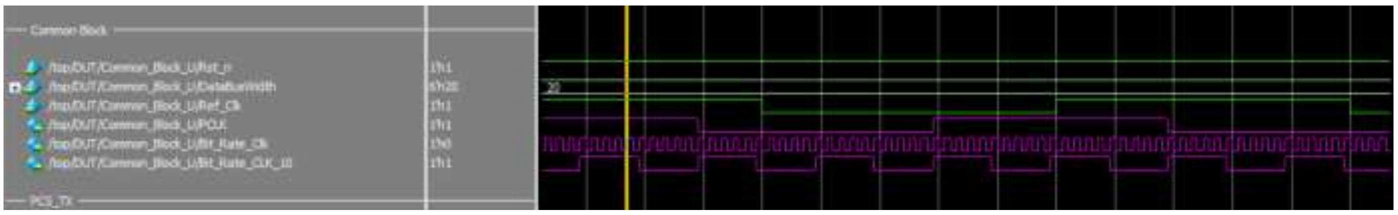
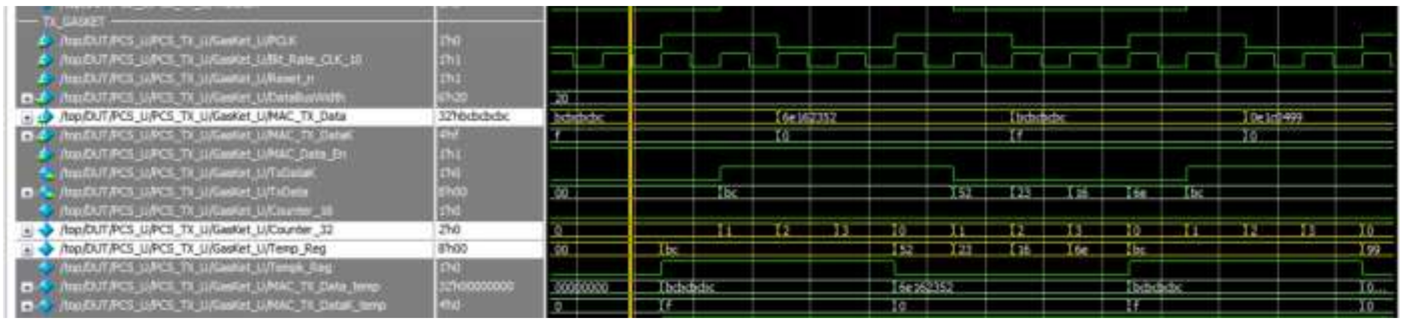

Common Block
/top/DUT/Common_Block_U/Rst_n
/top/DUT/Common_Block_U/DataBusWidth
/top/DUT/Common_Block_U/Ref_Clk
/top/DUT/Common_Block_U/PCLK
/top/DUT/Common_Block_U/Bit_Rate_Clk
/top/DUT/Common_Block_U/Bit_Rate_CLK_10
PCS_TX

Figure 36 Width = 8 bit clock generated

- For the (Width = 8) bit the Clocks generated from the Common Block and extracted from (Ref_CLk = 100MHz):

- *PCLK running at 500MHz.*
- *Symbol_Clk running at 500MHz.*
- *Bit_Rate_Clk running at 5GHz.*

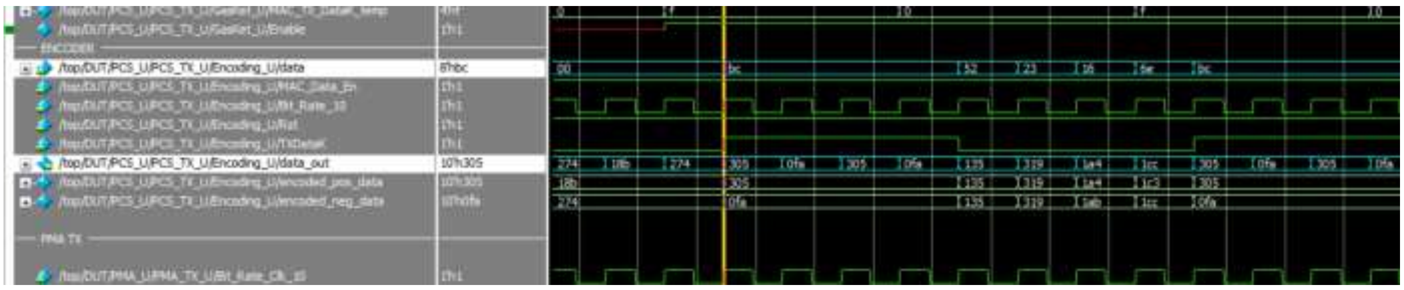
- <u>Gasket block dividing Data based on the Bus width</u>


TX_GASKET
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/PCLK
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Bit_Rate_CLK_10
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Reset_n
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/DataBusWidth
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/MAC_TX_Data
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/MAC_TX_DataK
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/MAC_Data_En
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/TxDataK
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/TxData
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Counter_10
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Counter_32
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Temp_Reg
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/Tempk_Reg
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/MAC_TX_Data_temp
/top/DUT/PCS_U/PCS_TX_U/Gasket_U/MAC_TX_DataK_temp

- for (width = 32)

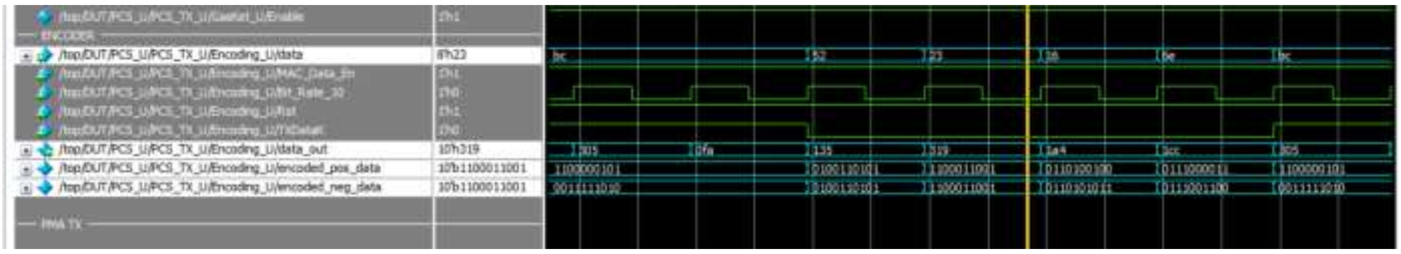
- *Max_Tx_Data = BCBCBCBC here is divided into 4 blocks 8-bit each and transmitted to the Encoder BC per block.*
- *Max_Tx_Data = 6e162325 here is divided into 4 blocks 8-bit each and transmitted to the Encoder.*

- Encoder block Encoded Data based on the disparity



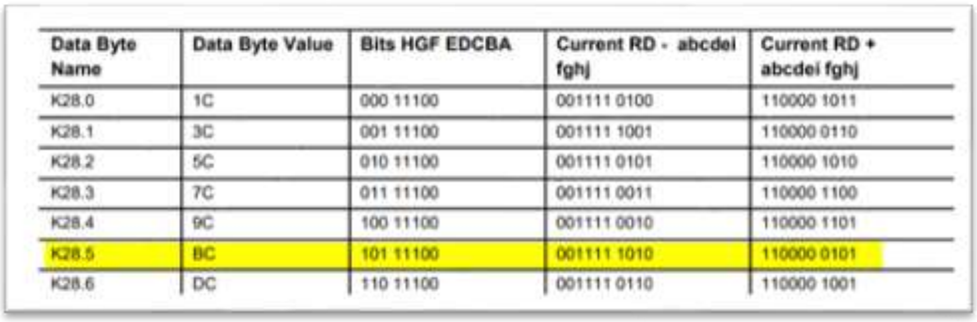
/top/DUT/PCS_U/PCS_TX_U/GasKet_U/Enable
ENCODER
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/data
8'hbc
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/MAC_Data_En
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/Bit_Rate_10
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/Rst
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/TXDataK
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/data_out
10'h305
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/encoded_pos_data
10'h305
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/encoded_neg_data
10'h0fa
PMA TX
/top/DUT/PMA_U/PMA_TX_U/Bit_Rate_Clk_10

Figure 37 balance between 0's and 1's

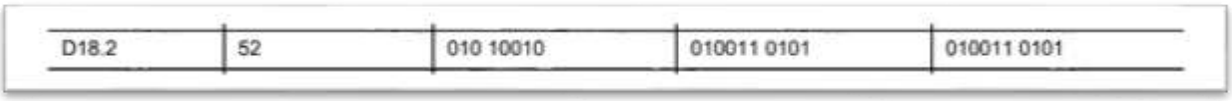


/top/DUT/PCS_U/PCS_TX_U/GasKet_U/Enable
ENCODER
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/data
8'h23
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/MAC_Data_En
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/Bit_Rate_10
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/Rst
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/TXDataK
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/data_out
10'h319
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/encoded_pos_data
10'b1100011001
/top/DUT/PCS_U/PCS_TX_U/Encoding_U/encoded_neg_data
10'b1100011001
PMA TX

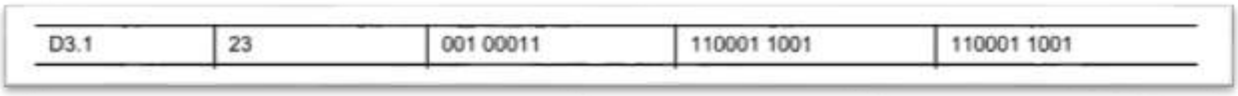
- Encoder Encoded each 8-bit and produce 10-bit data symbol with symbol clk rate for the appropriate disparity keeping the balance in 0's and 1's based on the table of disparity as in the figures



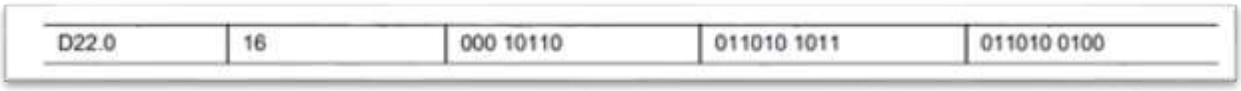
Data Byte Name	Data Byte Value	Bits HGF EDCBA	Current RD - abcdei fghj	Current RD + abcdei fghj
K28.0	1C	000 11100	001111 0100	110000 1011
K28.1	3C	001 11100	001111 1001	110000 0110
K28.2	5C	010 11100	001111 0101	110000 1010
K28.3	7C	011 11100	001111 0011	110000 1100
K28.4	9C	100 11100	001111 0010	110000 1101
K28.5	BC	101 11100	001111 1010	110000 0101
K28.6	DC	110 11100	001111 0110	110000 1001



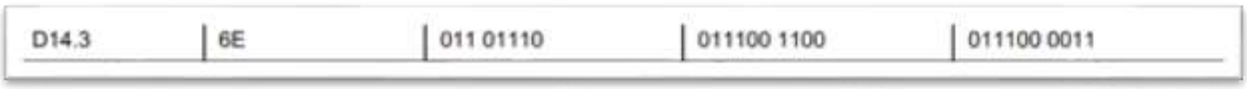
D18.2	52	010 10010	010011 0101	010011 0101



D3.1	23	001 00011	110001 1001	110001 1001

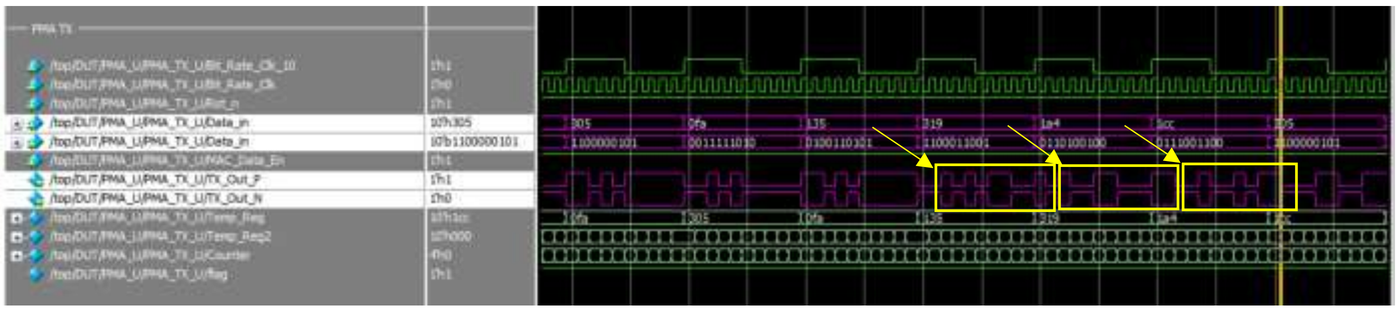


D22.0	16	000 10110	011010 1011	011010 0100



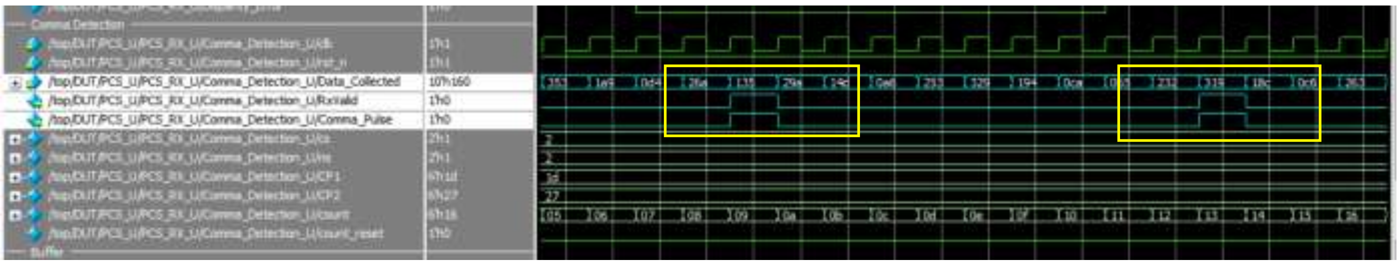
D14.3	6E	011 01110	011100 1100	011100 0011

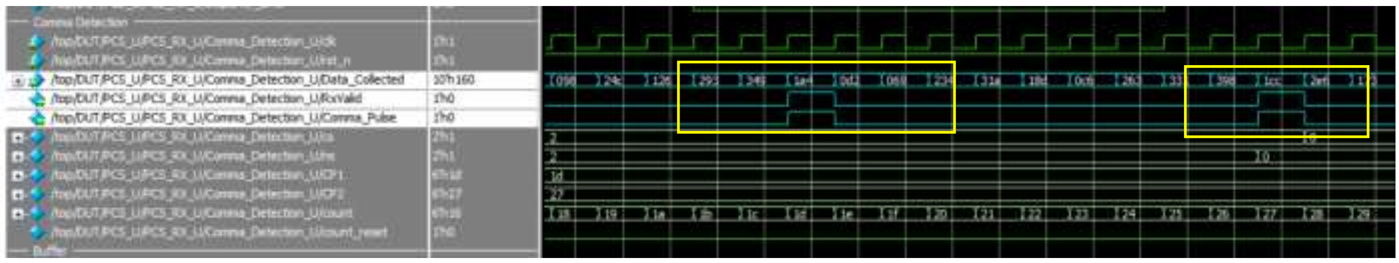
- PMA_TX P2S serializes the symbolled data

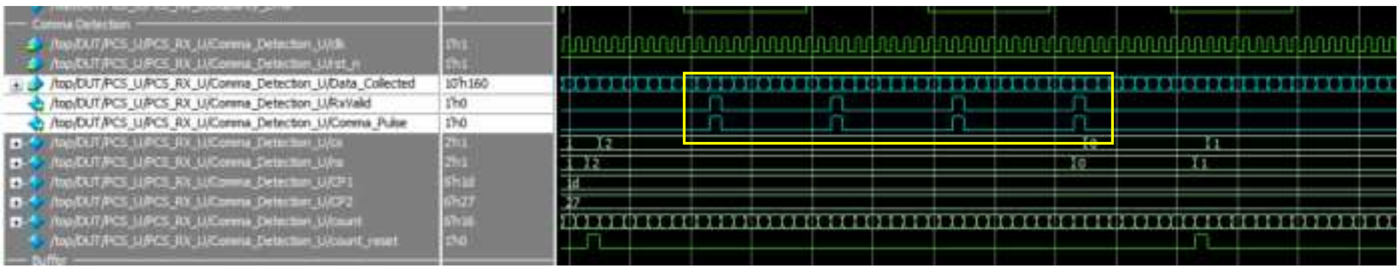

PMA TX
/top/DUT/PMA_U/PMA_TX_U/Bit_Rate_Clk_10
/top/DUT/PMA_U/PMA_TX_U/Bit_Rate_Clk
/top/DUT/PMA_U/PMA_TX_U/Rst_n
/top/DUT/PMA_U/PMA_TX_U/Data_in
/top/DUT/PMA_U/PMA_TX_U/Data_in
/top/DUT/PMA_U/PMA_TX_U/MAC_Data_En
/top/DUT/PMA_U/PMA_TX_U/TX_Out_P
/top/DUT/PMA_U/PMA_TX_U/TX_Out_N
/top/DUT/PMA_U/PMA_TX_U/Temp_Reg
/top/DUT/PMA_U/PMA_TX_U/Temp_Reg2
/top/DUT/PMA_U/PMA_TX_U/Counter
/top/DUT/PMA_U/PMA_TX_U/flag

- Transmitter serializes the parallel symbol data here for 10 Bit_Rate_Clk cycles from LSB to MSB.

- PMA_RX S2P deserializes the symbolled data

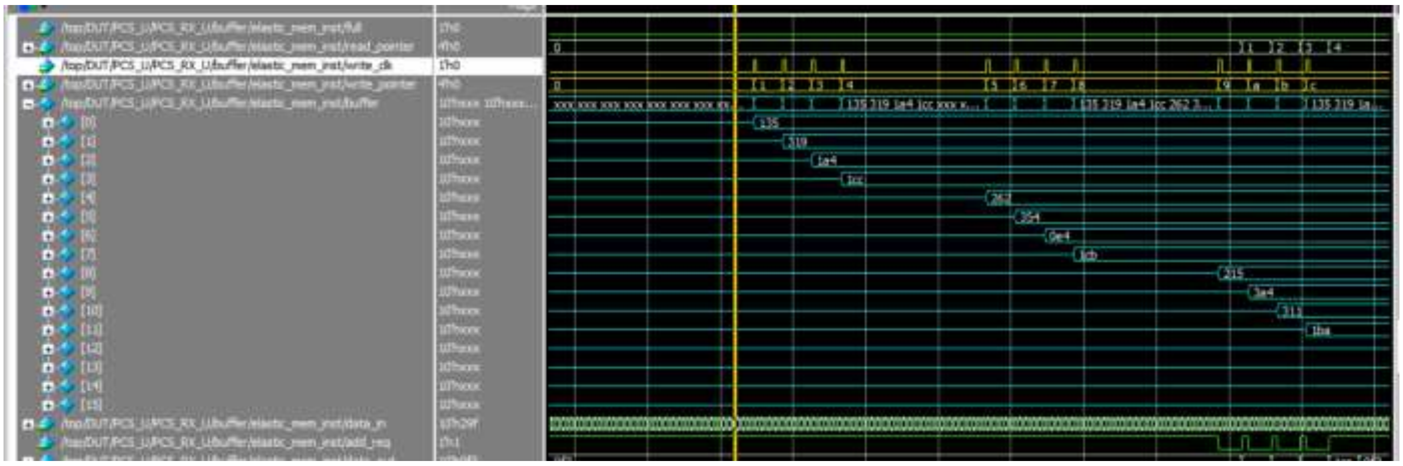

Comma Detection
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/clk
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/rst_n
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Data_Collected
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/RxValid
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Comma_Pulse
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/cs
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/ns
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP1
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP2
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count_reset
Buffer


Comma Detection
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/clk
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/rst_n
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Data_Collected
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/RxValid
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Comma_Pulse
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/cs
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/ns
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP1
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP2
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count_reset
Buffer


Comma Detection
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/clk
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/rst_n
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Data_Collected
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/RxValid
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/Comma_Pulse
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/cs
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/ns
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP1
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/CP2
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count_reset
Buffer

- After date serialized from the TX the receiver collects the data in S2P (Serial to Parallel) when collecting COMMA Order Set It raise the Rx_Valid flag indicating the Symbol lock as we see here it collects the serialized data converting them to parallel data sending them to next stage Elastic Buffer.

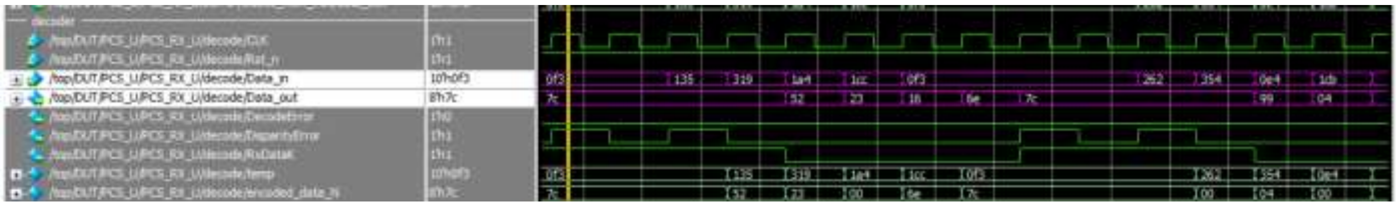
- <u>Elastic Buffer Storing The Data</u>



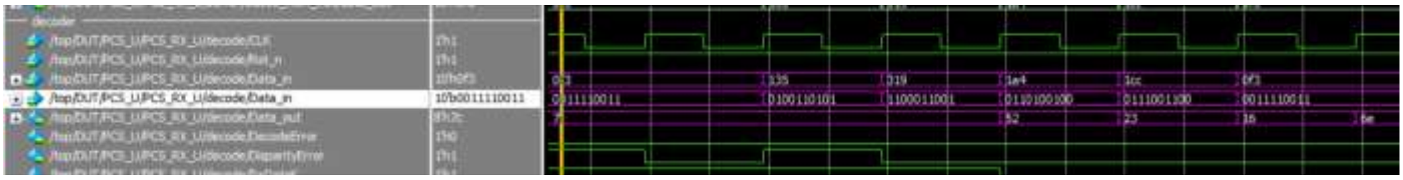
```
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/full           1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/read_pointer   4'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/write_clk      1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/write_pointer  4'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/buffer         10'hxxx 10'hxxx...
  [0]   10'hxxx
  [1]   10'hxxx
  [2]   10'hxxx
  [3]   10'hxxx
  [4]   10'hxxx
  [5]   10'hxxx
  [6]   10'hxxx
  [7]   10'hxxx
  [8]   10'hxxx
  [9]   10'hxxx
  [10]  10'hxxx
  [11]  10'hxxx
  [12]  10'hxxx
  [13]  10'hxxx
  [14]  10'hxxx
  [15]  10'hxxx
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/data_in        10'h29f
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/add_req        1'h1
```

-With The K28 Comma Detection Lock the data are stored inside the Elastic buffer as we see here, therefore an increasing in write pointer with each write operation happens.

- <u>Decoding the symbol encoded data into the original Data</u>

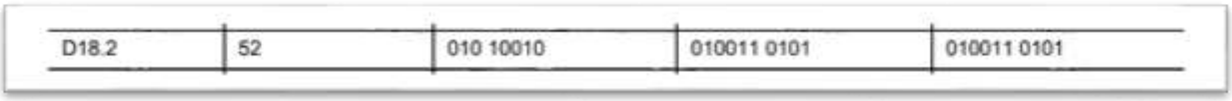


```
decoder
/top/DUT/PCS_U/PCS_RX_U/decode/CLK             1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Rst_n           1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in         10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/Data_out        8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError     1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DisparityError  1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK         1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/temp            10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_in 8'h7c
```

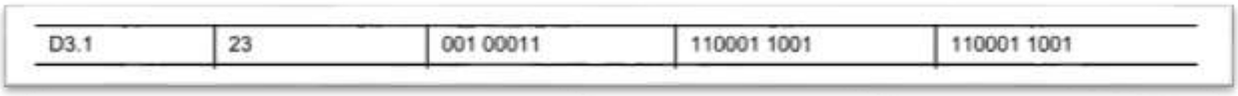


```
decoder
/top/DUT/PCS_U/PCS_RX_U/decode/CLK             1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Rst_n           1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in         10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in         10'b0011110011
/top/DUT/PCS_U/PCS_RX_U/decode/Data_out        8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError     1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DisparityError  1'h1
```

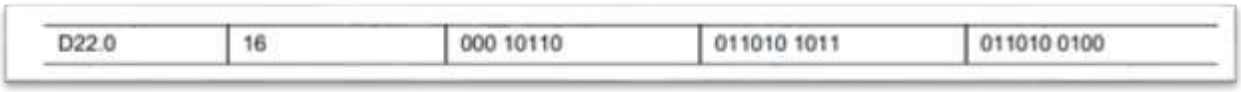
Decoder takes the symbol encoded data converting them into 8-bit original data:



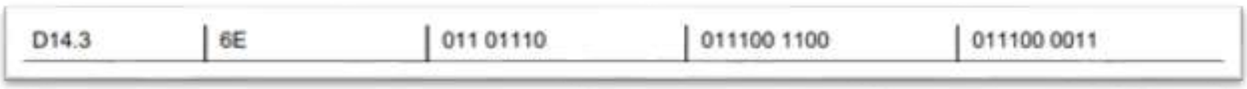
D18.2	52	010 10010	010011 0101	010011 0101



D3.1	23	001 00011	110001 1001	110001 1001

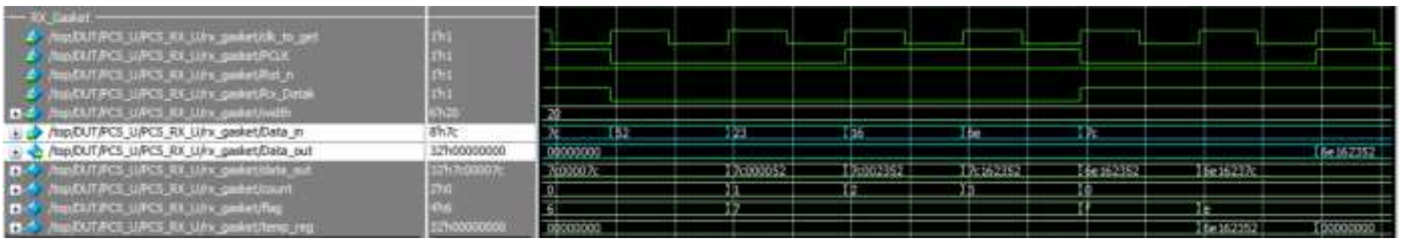


D22.0	16	000 10110	011010 1011	011010 0100

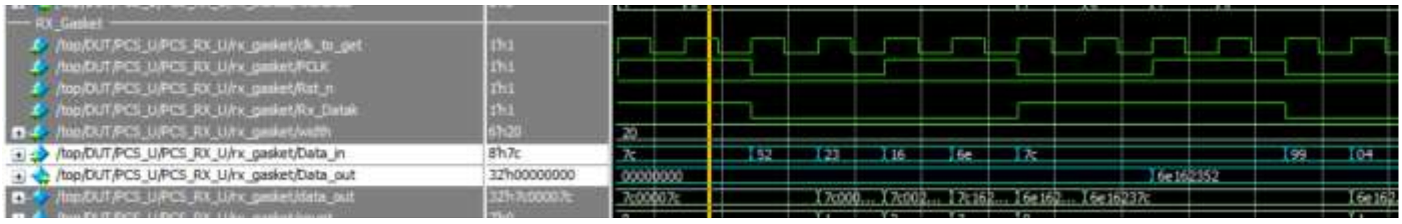


D14.3	6E	011 01110	011100 1100	011100 0011

- <u>Rx_Gasket Collecting the data to Rx output</u>



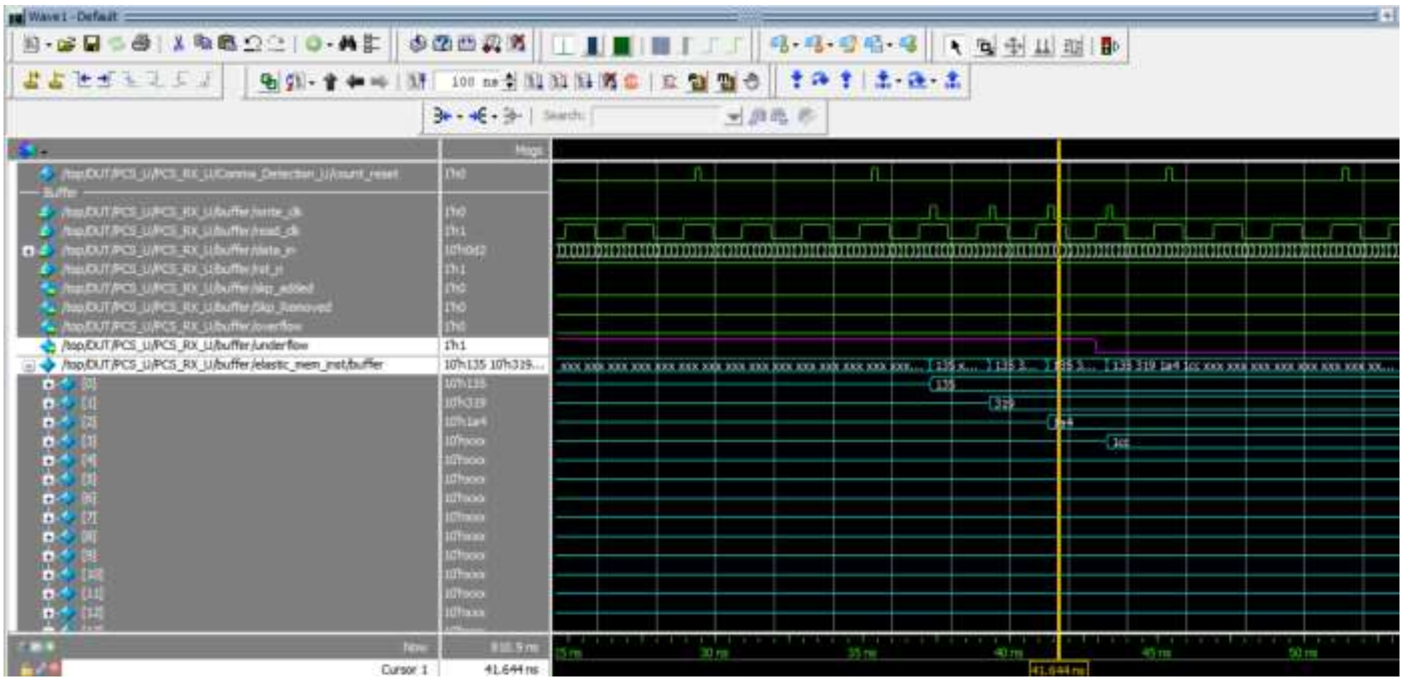
```
RX_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get    1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK          1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n         1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak      1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width         6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in       8'h7c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_out      32'h00000000
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/data_out      32'h7c00007c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/count         2'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/flag          4'h6
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/temp_reg      32'h00000000
```



```
RX_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get    1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK          1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n         1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak      1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width         6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in       8'h7c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_out      32'h00000000
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/data_out      32'h7c00007c
```

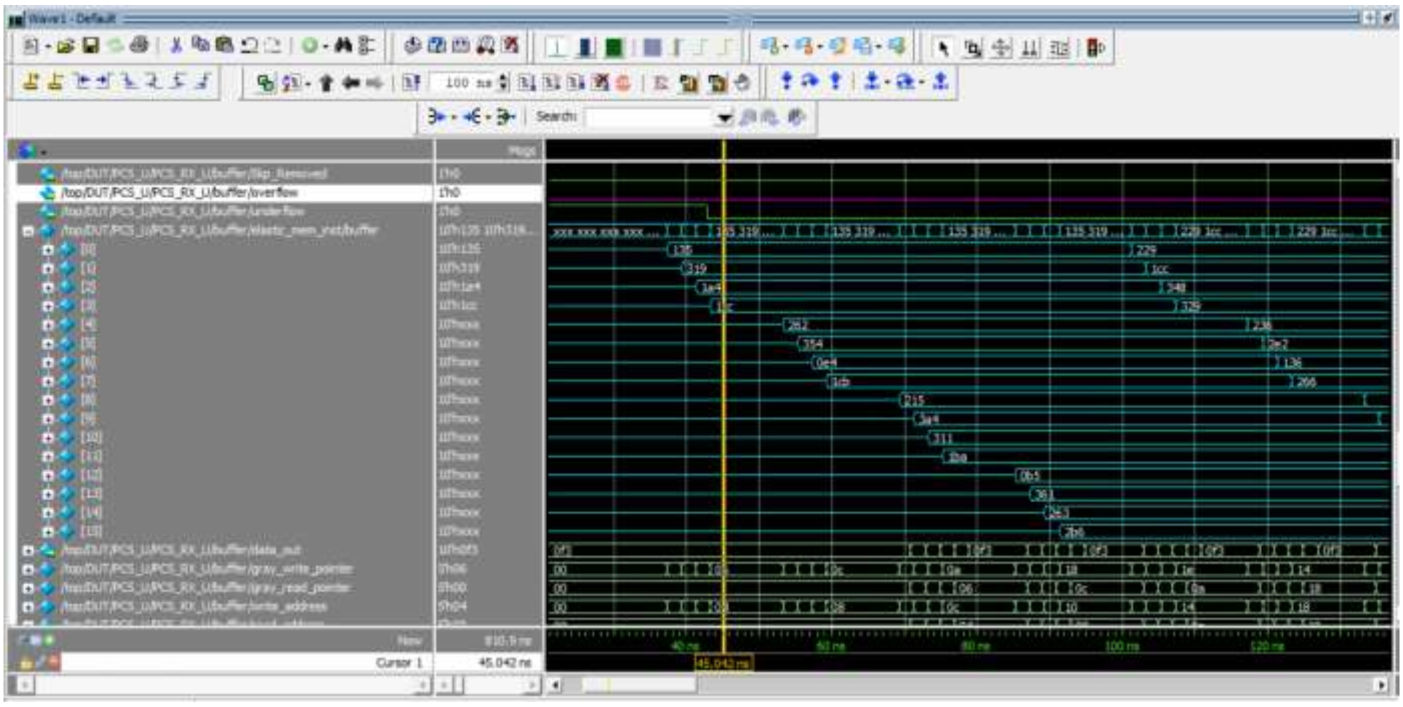
Rx_gasket collects data into 32-bit original data bus width to the output.

- <u>Elastic Buffer Underflow</u>

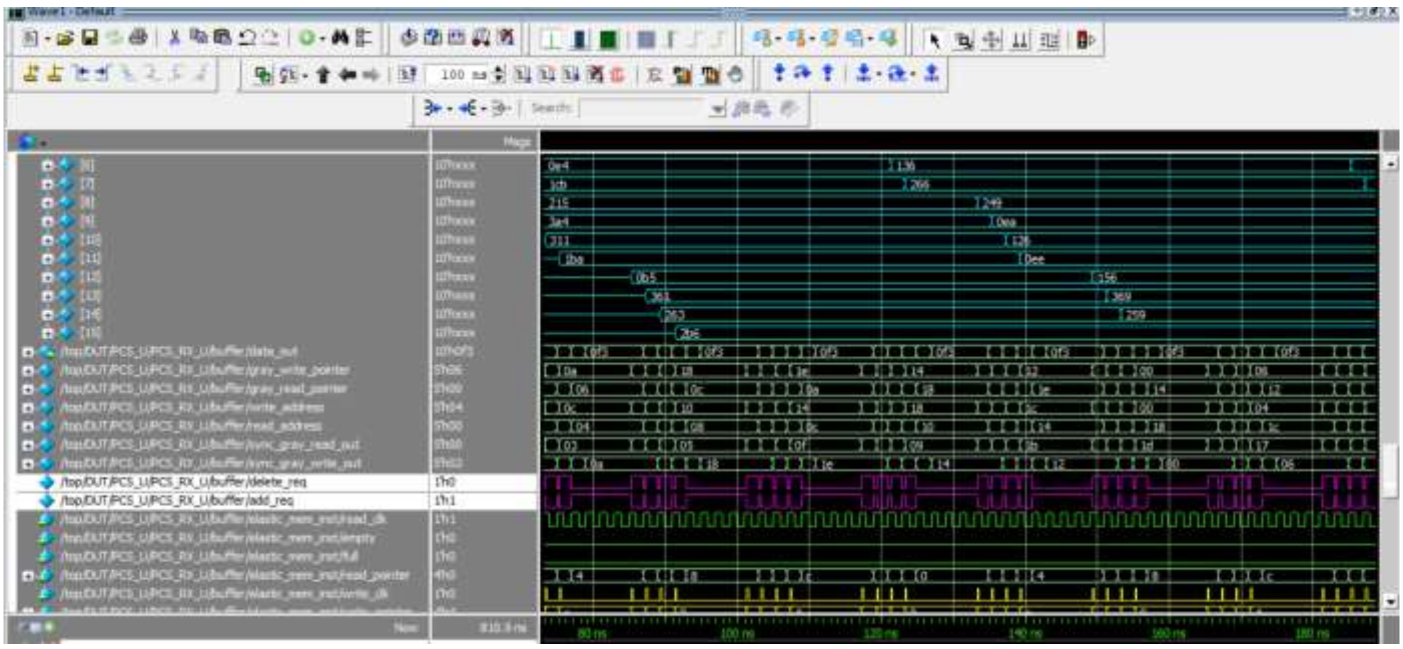


```
Wave1 - Default
                                                                Msgs
/top/DUT/PCS_U/PCS_RX_U/Comma_Detection_U/count_reset           1'h0
Buffer
/top/DUT/PCS_U/PCS_RX_U/buffer/write_clk                        1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/read_clk                         1'h1
/top/DUT/PCS_U/PCS_RX_U/buffer/data_in                          10'h0d2
/top/DUT/PCS_U/PCS_RX_U/buffer/rst_n                            1'h1
/top/DUT/PCS_U/PCS_RX_U/buffer/skp_added                        1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/Skp_Removed                      1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/overflow                         1'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/underflow                        1'h1
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/buffer          10'h135 10'h319...
  [0]                                                           10'h135
  [1]                                                           10'h319
  [2]                                                           10'h1a4
  [3]                                                           10'hxxx
  [4]                                                           10'hxxx
  [5]                                                           10'hxxx
  [6]                                                           10'hxxx
  [7]                                                           10'hxxx
  [8]                                                           10'hxxx
  [9]                                                           10'hxxx
  [10]                                                          10'hxxx
  [11]                                                          10'hxxx
  [12]                                                          10'hxxx
                                                    Now         810.9 ns
                                                    Cursor 1    41.644 ns
```

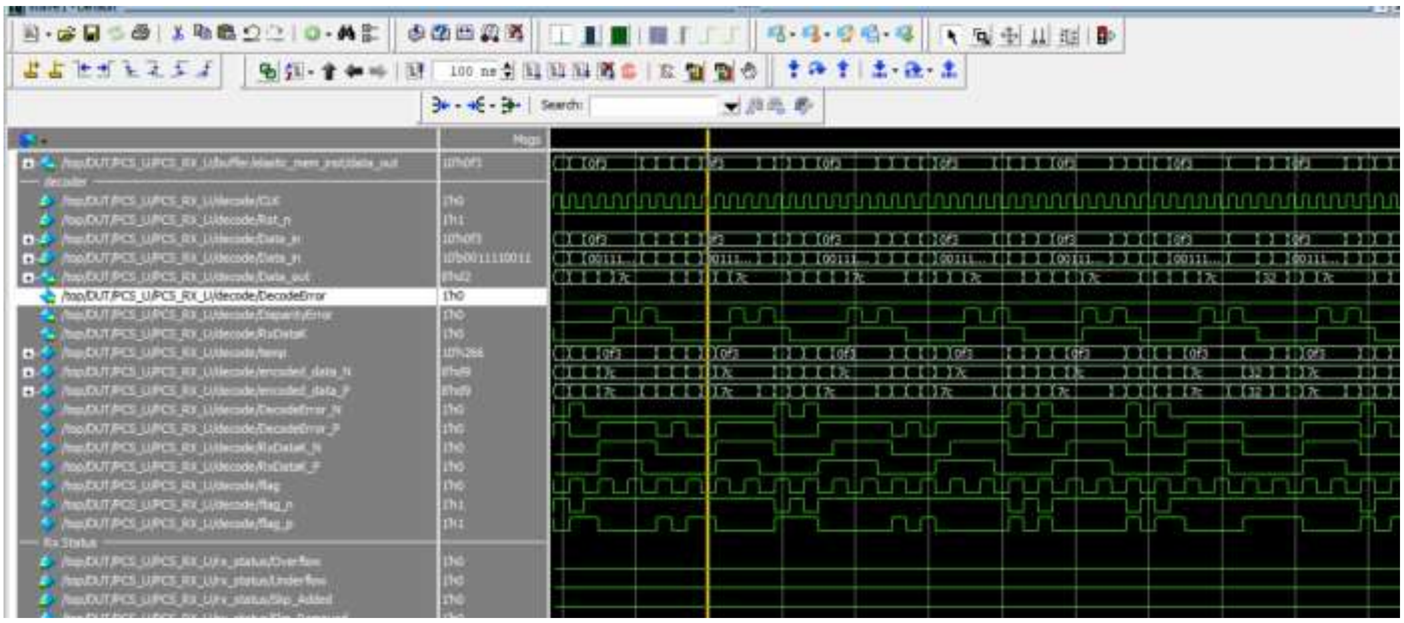
- Elastic Buffer Overflow


Wave1 - Default
/top/DUT/PCS_U/PCS_RX_U/buffer/Skip_Removed
/top/DUT/PCS_U/PCS_RX_U/buffer/overflow
/top/DUT/PCS_U/PCS_RX_U/buffer/underflow
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/buffer
/top/DUT/PCS_U/PCS_RX_U/buffer/data_out
/top/DUT/PCS_U/PCS_RX_U/buffer/gray_write_pointer
/top/DUT/PCS_U/PCS_RX_U/buffer/gray_read_pointer
/top/DUT/PCS_U/PCS_RX_U/buffer/write_address
Now
Cursor 1
45.042 ns

- Elastic Buffer (Threshold Monitor) Add Request – Delete Requests

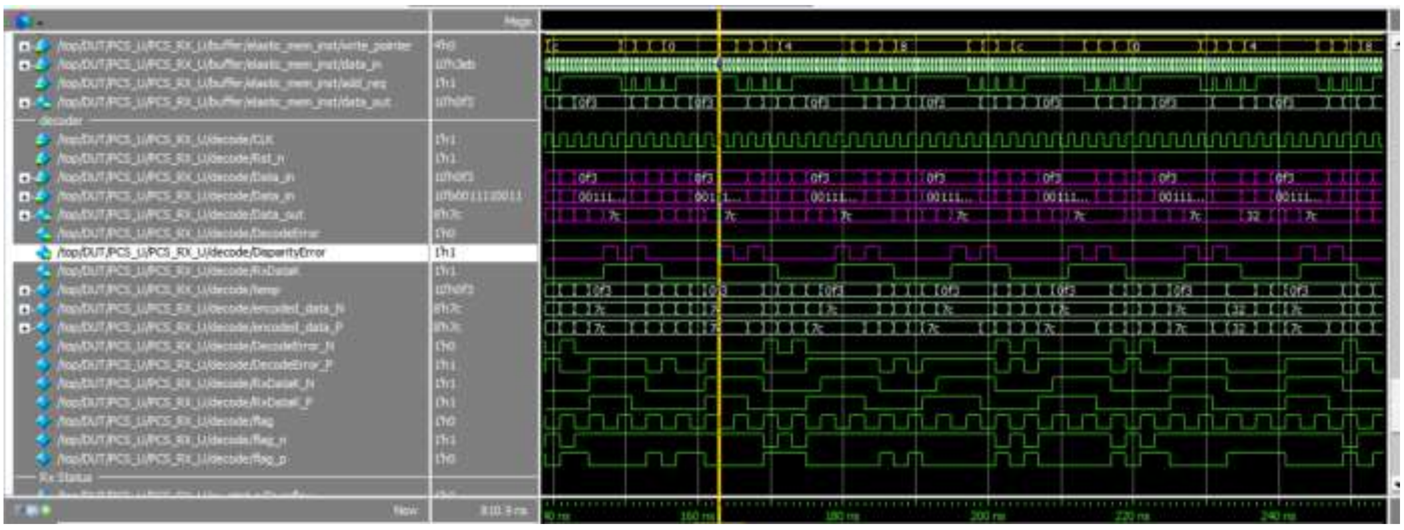

Wave1 - Default
/top/DUT/PCS_U/PCS_RX_U/buffer/data_out
/top/DUT/PCS_U/PCS_RX_U/buffer/gray_write_pointer
/top/DUT/PCS_U/PCS_RX_U/buffer/gray_read_pointer
/top/DUT/PCS_U/PCS_RX_U/buffer/write_address
/top/DUT/PCS_U/PCS_RX_U/buffer/read_address
/top/DUT/PCS_U/PCS_RX_U/buffer/sync_gray_read_out
/top/DUT/PCS_U/PCS_RX_U/buffer/sync_gray_write_out
/top/DUT/PCS_U/PCS_RX_U/buffer/delete_req
/top/DUT/PCS_U/PCS_RX_U/buffer/add_req
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/read_clk
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/empty
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/full
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/read_pointer
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/write_clk
Now

- Decoding Error



```
Msgs
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/data_out    10'h0f1
decoder
/top/DUT/PCS_U/PCS_RX_U/decode/CLK                          1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/Rst_n                        1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in                      10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in                      10'b0011110011
/top/DUT/PCS_U/PCS_RX_U/decode/Data_out                     8'hd2
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError                  1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DisparityError               1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK                      1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/temp                         10'h266
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_N               8'hd9
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P               8'hd9
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N                1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P                1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N                    1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P                    1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag                         1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n                       1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p                       1'h1
Rx Status
/top/DUT/PCS_U/PCS_RX_U/rx_status/Overflow                  1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Underflow                 1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Added                 1'h0
```

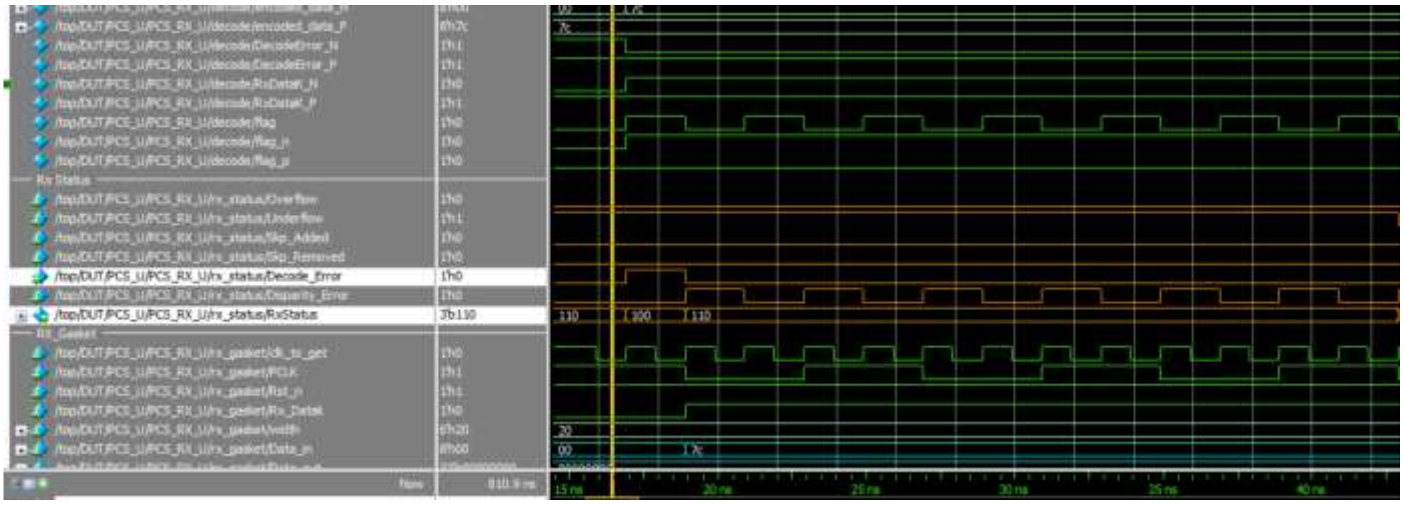
- Decoder Disparity Error



```
Msgs
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/write_pointer   4'h0
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/data_in         10'h3eb
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/add_req         1'h1
/top/DUT/PCS_U/PCS_RX_U/buffer/elastic_mem_inst/data_out        10'h0f3
decoder
/top/DUT/PCS_U/PCS_RX_U/decode/CLK                              1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Rst_n                            1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in                          10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/Data_in                          10'b0011110011
/top/DUT/PCS_U/PCS_RX_U/decode/Data_out                         8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError                      1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DisparityError                   1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK                          1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/temp                             10'h0f3
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_N                   8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P                   8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N                    1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P                    1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N                        1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P                        1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag                             1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n                           1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p                           1'h0
Rx Status
Now                                                             810.9 ns
```

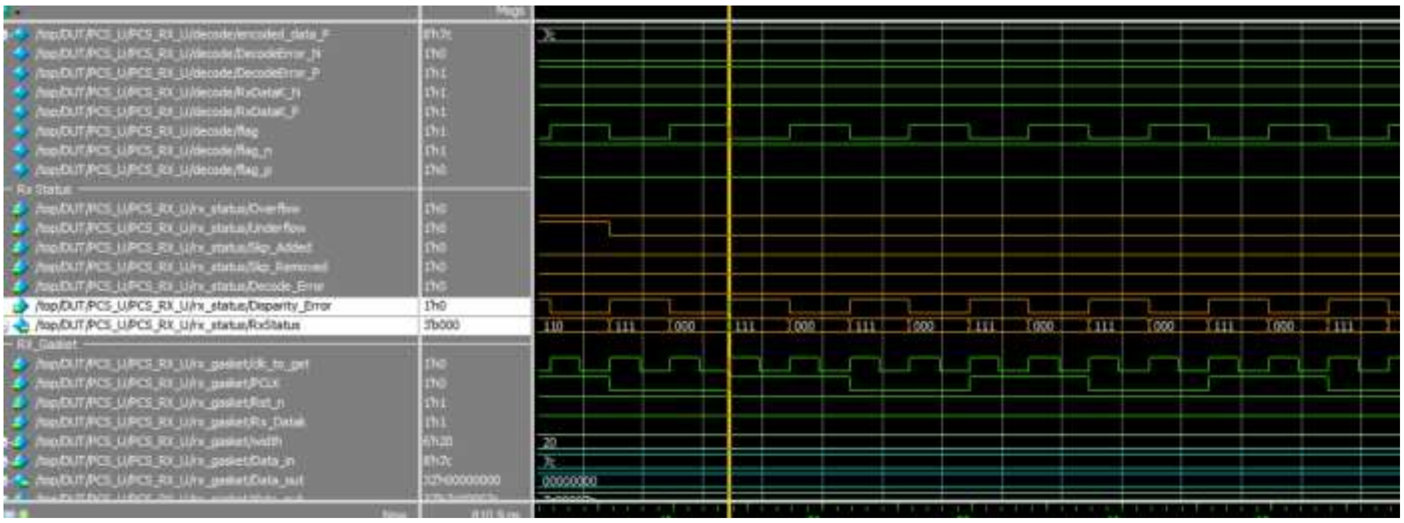
Two positive or negative encoded data comes in sequence without toggling disparity due to end padding Command while buffer under threshold waiting for collecting 4 symbols of data(acceptable).

- Decoding Error - Rx_status.



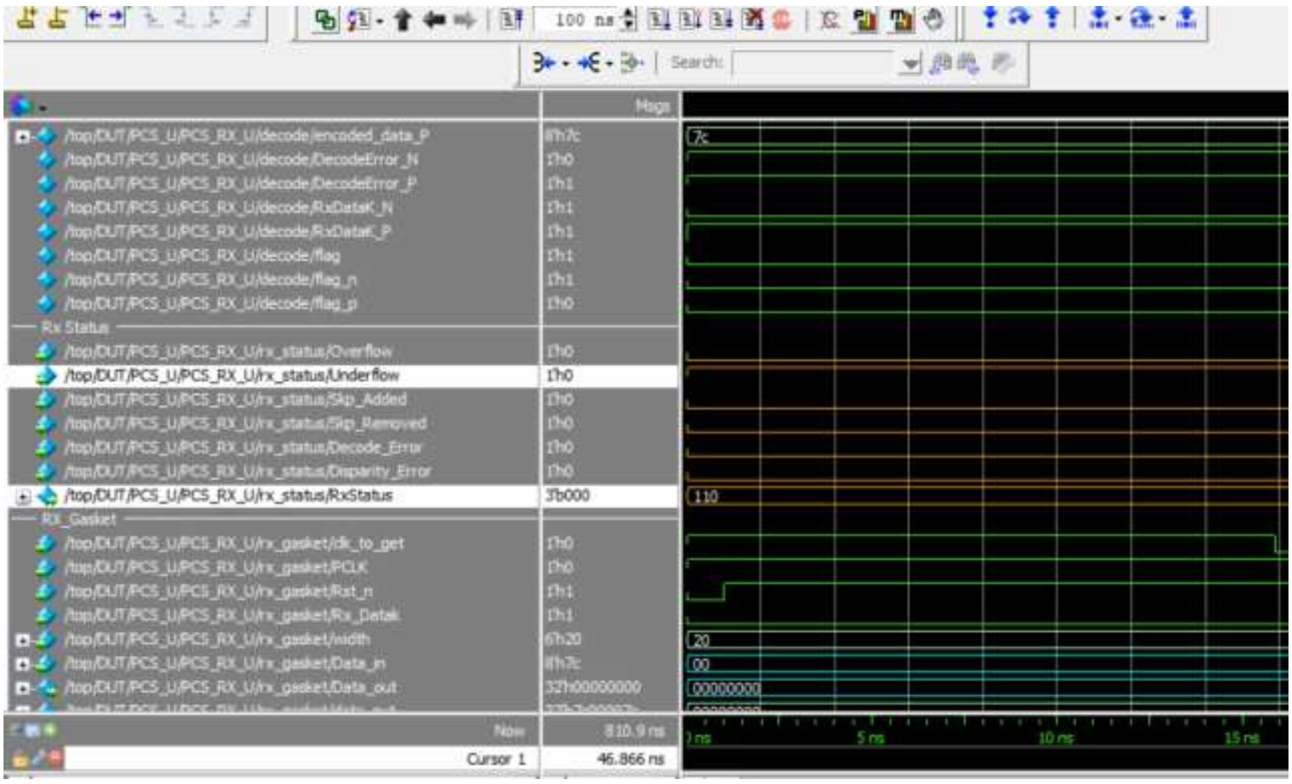
```
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P     8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N      1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P      1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N          1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P          1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag               1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n             1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p             1'h0
Rx Status
/top/DUT/PCS_U/PCS_RX_U/rx_status/Overflow        1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Underflow       1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Added       1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Removed     1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Decode_Error    1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Disparity_Error 1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/RxStatus        3'b110
Rx_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get      1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK            1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n           1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak        1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width           6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in         8'h00
Now                                               810.9 ns
```

- Disparity Error - Rx_status.



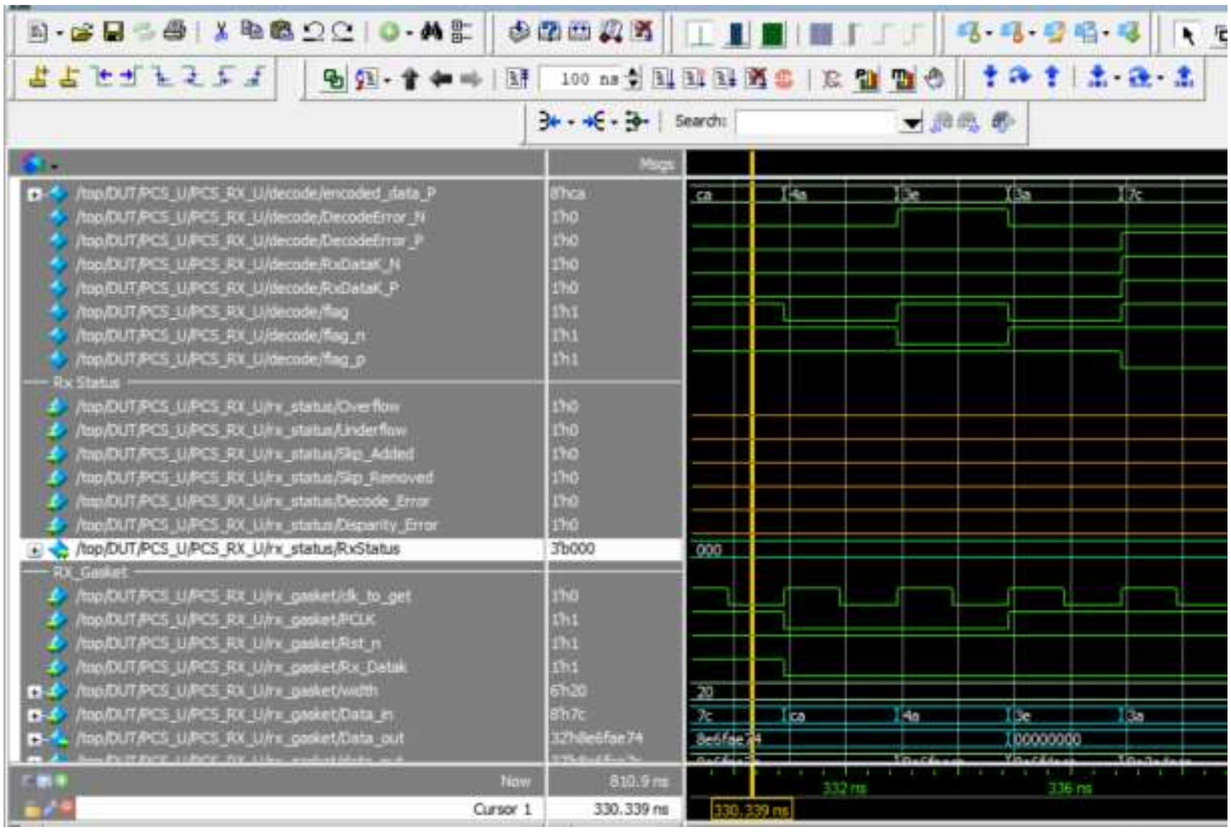
```
Msgs
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P     8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N      1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P      1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N          1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P          1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag               1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n             1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p             1'h0
Rx Status
/top/DUT/PCS_U/PCS_RX_U/rx_status/Overflow        1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Underflow       1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Added       1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Removed     1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Decode_Error    1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Disparity_Error 1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/RxStatus        3'b000
Rx_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get      1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK            1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n           1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak        1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width           6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in         8'h7c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_out        32'h00000000
Now                                               810.9 ns
```

- Underflow- Rx_status.



```
                                                     Msgs
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P        8'h7c
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N         1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P         1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N             1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P             1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag                  1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n                1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p                1'h0
Rx Status
/top/DUT/PCS_U/PCS_RX_U/rx_status/Overflow           1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Underflow          1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Added          1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Removed        1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Decode_Error       1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Disparity_Error    1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/RxStatus           3'b000
RX_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get         1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK               1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n              1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak           1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width              6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in            8'h7c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_out           32'h00000000
                                            Now      810.9 ns
                                       Cursor 1      46.866 ns
```

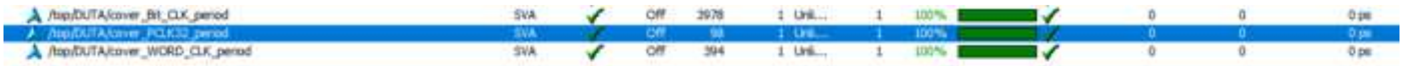
- Data is OK- Rx_status.



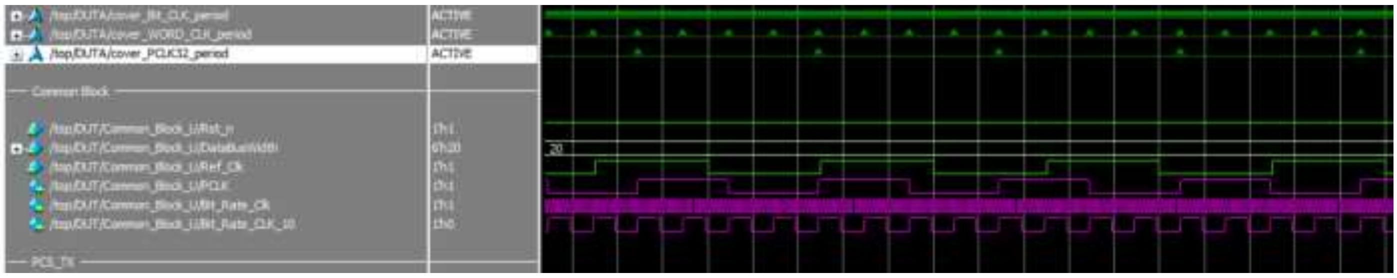
```
                                                     Msgs
/top/DUT/PCS_U/PCS_RX_U/decode/encoded_data_P        8'hca
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_N         1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/DecodeError_P         1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_N             1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/RxDataK_P             1'h0
/top/DUT/PCS_U/PCS_RX_U/decode/flag                  1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_n                1'h1
/top/DUT/PCS_U/PCS_RX_U/decode/flag_p                1'h1
Rx Status
/top/DUT/PCS_U/PCS_RX_U/rx_status/Overflow           1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Underflow          1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Added          1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Skp_Removed        1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Decode_Error       1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/Disparity_Error    1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_status/RxStatus           3'b000
RX_Gasket
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/clk_to_get         1'h0
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/PCLK               1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rst_n              1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Rx_Datak           1'h1
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/width              6'h20
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_in            8'h7c
/top/DUT/PCS_U/PCS_RX_U/rx_gasket/Data_out           32'h8e6fae74
                                            Now      810.9 ns
                                       Cursor 1      330.339 ns
```

CLK Periods

-DataBusWidth = 32

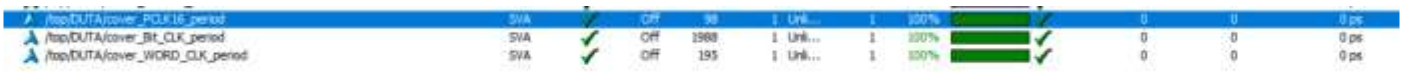


```
/top/DUTA/cover_Bit_CLK_period     SVA  ✓  Off  3978  1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_PCLK32_period      SVA  ✓  Off    98  1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_WORD_CLK_period    SVA  ✓  Off   394  1 Unli...  1  100%  ✓  0  0  0 ps
```

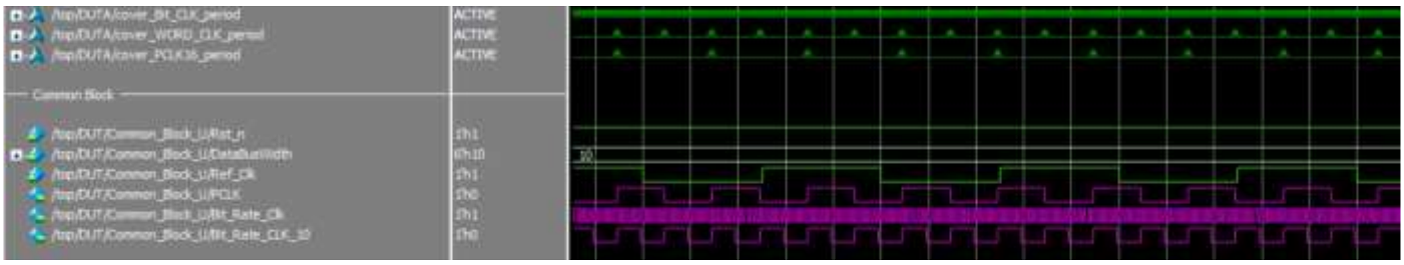


```
/top/DUTA/cover_Bit_CLK_period                 ACTIVE
/top/DUTA/cover_WORD_CLK_period                ACTIVE
/top/DUTA/cover_PCLK32_period                  ACTIVE
Common Block
/top/DUT/Common_Block_U/Rst_n                  1'h1
/top/DUT/Common_Block_U/DataBusWidth           6'h20
/top/DUT/Common_Block_U/Ref_Clk                1'h1
/top/DUT/Common_Block_U/PCLK                   1'h1
/top/DUT/Common_Block_U/Bit_Rate_Clk           1'h1
/top/DUT/Common_Block_U/Bit_Rate_CLK_10        1'h0
PCS_TX
```

-DataBusWidth = 16

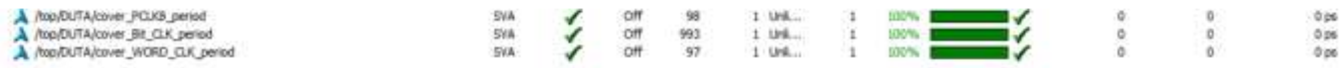


```
/top/DUTA/cover_PCLK16_period      SVA  ✓  Off    98  1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_Bit_CLK_period     SVA  ✓  Off  1988  1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_WORD_CLK_period    SVA  ✓  Off   195  1 Unli...  1  100%  ✓  0  0  0 ps
```

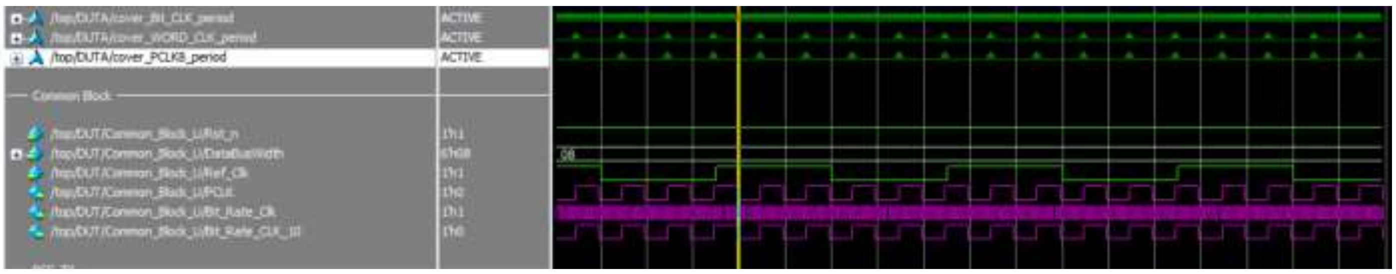


```
/top/DUTA/cover_Bit_CLK_period                 ACTIVE
/top/DUTA/cover_WORD_CLK_period                ACTIVE
/top/DUTA/cover_PCLK16_period                  ACTIVE
Common Block
/top/DUT/Common_Block_U/Rst_n                  1'h1
/top/DUT/Common_Block_U/DataBusWidth           6'h10
/top/DUT/Common_Block_U/Ref_Clk                1'h1
/top/DUT/Common_Block_U/PCLK                   1'h0
/top/DUT/Common_Block_U/Bit_Rate_Clk           1'h1
/top/DUT/Common_Block_U/Bit_Rate_CLK_10        1'h0
```

-DataBusWidth = 8



```
/top/DUTA/cover_PCLK8_period      SVA  ✓  Off  98   1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_Bit_CLK_period    SVA  ✓  Off  993  1 Unli...  1  100%  ✓  0  0  0 ps
/top/DUTA/cover_WORD_CLK_period   SVA  ✓  Off  97   1 Unli...  1  100%  ✓  0  0  0 ps
```



```
/top/DUTA/cover_Bit_CLK_period                 ACTIVE
/top/DUTA/cover_WORD_CLK_period                ACTIVE
/top/DUTA/cover_PCLK8_period                   ACTIVE
— Common Block —
/top/DUT/Common_Block_U/Rst_n                  1'h1
/top/DUT/Common_Block_U/DataBusWidth           6'h08
/top/DUT/Common_Block_U/Ref_Clk                1'h1
/top/DUT/Common_Block_U/PCLK                   1'h0
/top/DUT/Common_Block_U/Bit_Rate_Clk           1'h1
/top/DUT/Common_Block_U/Bit_Rate_CLK_III       1'h0
```

TX Verification

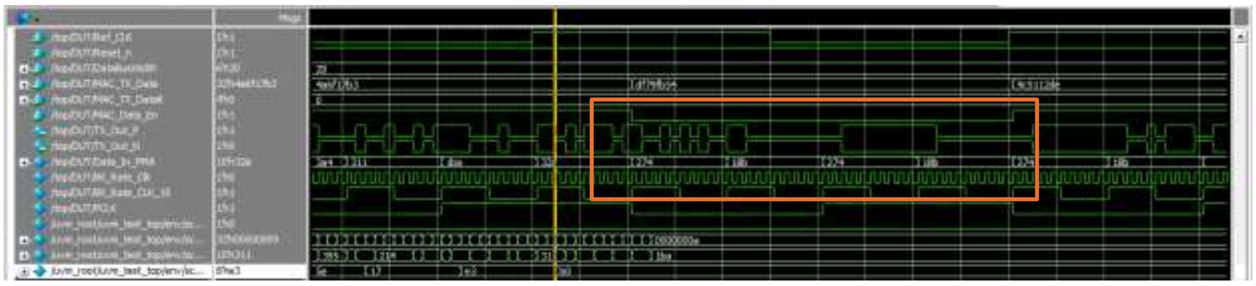
TX Test Strategy

Data_Bus_Width	MAC_En	MAC_Data	MAC_DataK	Rst_n	Test Feature
6’d8	1’b0	8’hxx	4’hx	1’b1	Check Mac Enable enforce all blocks to be disabled
6’d8	1’b1	8’hxx	4’hx	1’b0	Check active low reset Data to be Zeros
6’d8	1’b1	8’h26	4’h0	1’b1	Check **byte** to be sent as actual Data not Command
6’d8	1’b1	8’hBC	4’h1	1’b1	Check **byte** to be sent as Command “**COMMA**”
6’d16	1’b1	16’h719c	4’h0	1’b1	Check **2-bytes** to be sent as actual Data
6’d16	1’b1	16’h71BC	4’h1	1’b1	Check sent **2-bytes** as LSB byte sent as command and the other sent as Actual Data
6’d16	1’b1	16’hBC71	4’h2	1’b1	Check sent the MSB byte as command
6’d16	1’b1	16’hBCBC	4’h3	1’b1	Check **2-bytes** are sent as actual Data
6’d32	1’b1	32’h26c4dd81	4’h0	1’b1	Check the whole **word** is sent as Data
6’d32	1’b1	32’h26c4ddbc	4’h1	1’b1	Check that LSB Byte is sent as command
6’d32	1’b1	32’h26c4bc81	4’h2	1’b1	Check that 2nd Byte is sent as command
6’d32	1’b1	32’h26c4bcbc	4’h3	1’b1	Check that 1st & 2nd Byte is sent as command
6’d32	1’b1	32’h26bcdd81	4’h4	1’b1	Check that 3rd Byte is sent as command
6’d32	1’b1	32’h26bcddbc	4’h5	1’b1	Check that 1st & 3rd Byte is sent as command
6’d32	1’b1	32’h26bcbc81	4’h6	1’b1	Check that 2nd & 3rd Byte is sent as command

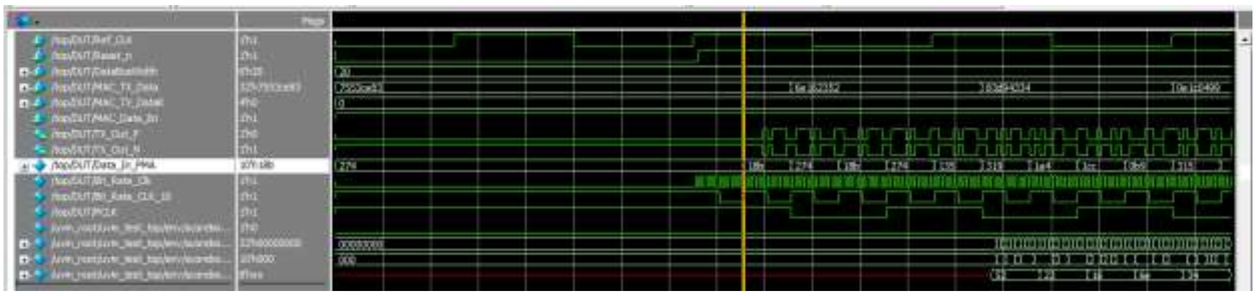
6’d32	1’b1	32’h26bcbcbc	4’h7	1’b1	Check that 1st ,2nd, 3rd Byte is sent as command
6’d32	1’b1	32’hbcc4dd81	4’h8	1’b1	Check that 4th Byte is sent as command
6’d32	1’b1	32’hbcc4ddbc	4’h9	1’b1	Check that 4th & 1st Byte is sent as command
6’d32	1’b1	32’hbcc4bc81	4’hA	1’b1	Check that 2nd & 4th Byte is sent as command
6’d32	1’b1	32’hbcc4bcbc	4’hB	1’b1	Check that 1st , 2nd ,4th Byte is sent as command
6’d32	1’b1	32’hbcbcdd81	4’hC	1’b1	Check that 3rd , 4th Byte is sent as command
6’d32	1’b1	32’hbcbcddbc	4’hD	1’b1	Check that 1st ,3rd ,4th Byte is sent as command
6’d32	1’b1	32’hbcbcbc81	4’hE	1’b1	Check that 2nd ,3rd ,4th Byte is sent as command
6’d32	1’b1	32’hbcbcbcbc	4’hF	1’b1	Check the whole word is sent as command

Features:

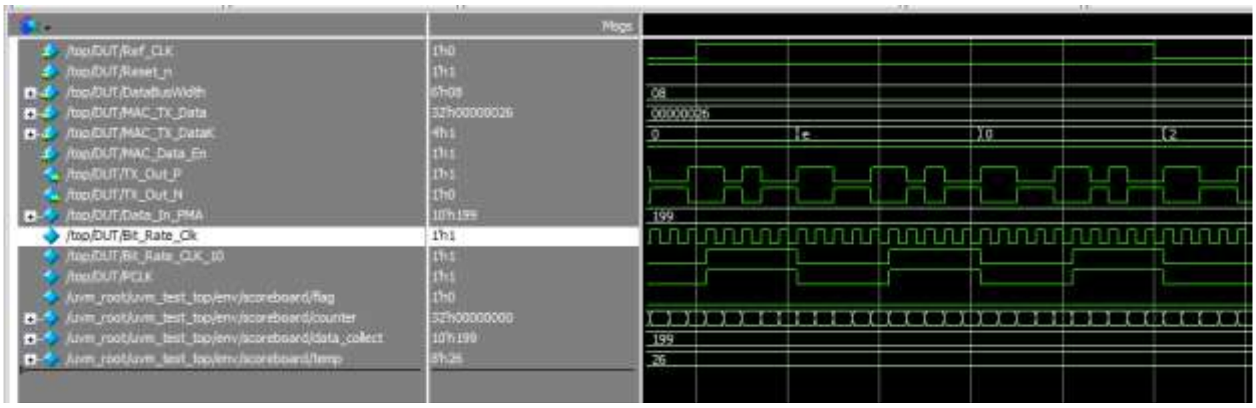
1. **Check data sent when MAC_DATA_En signal is de-activated (enable = 0)**



2. **Check active low reset to be Zeros**

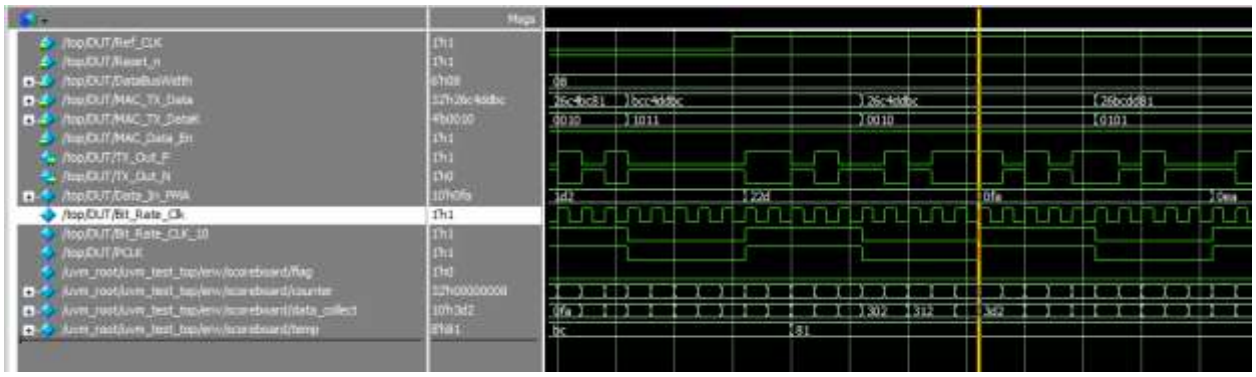


3. **Check data 8’h26 with width 8 and it’s actual data not a command**



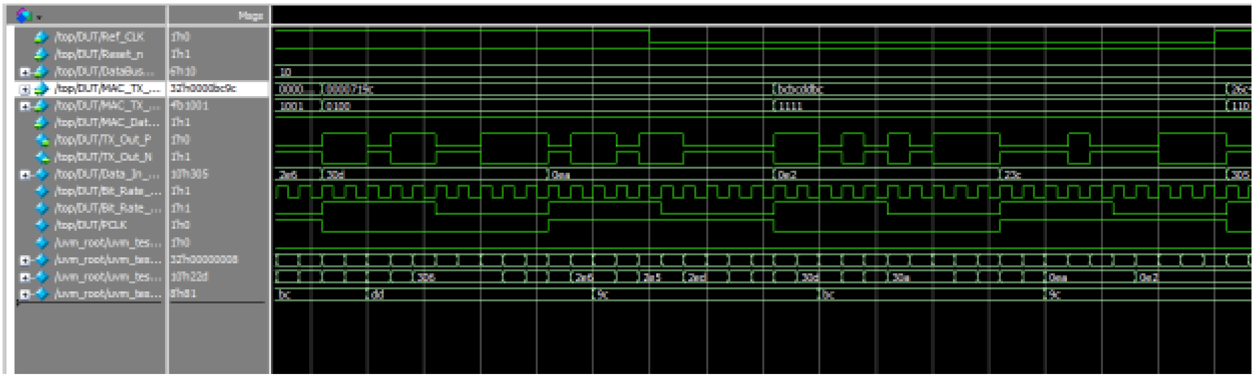
```
/top/DUT/Ref_CLK                                   1'h0
/top/DUT/Reset_n                                   1'h1
/top/DUT/DataBusWidth                              6'h08
/top/DUT/MAC_TX_Data                               32'h00000026
/top/DUT/MAC_TX_DataK                              4'h1
/top/DUT/MAC_Data_En                               1'h1
/top/DUT/TX_Out_P                                  1'h1
/top/DUT/TX_Out_N                                  1'h0
/top/DUT/Data_In_PMA                               10'h199
/top/DUT/Bit_Rate_Clk                              1'h1
/top/DUT/Bit_Rate_CLK_10                           1'h1
/top/DUT/PCLK                                      1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag         1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter      32'h00000000
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h199
/uvm_root/uvm_test_top/env/scoreboard/temp         8'h26
```

4. **Check data 8'hbc to be sent as command with encoded value = 10'h0fa**



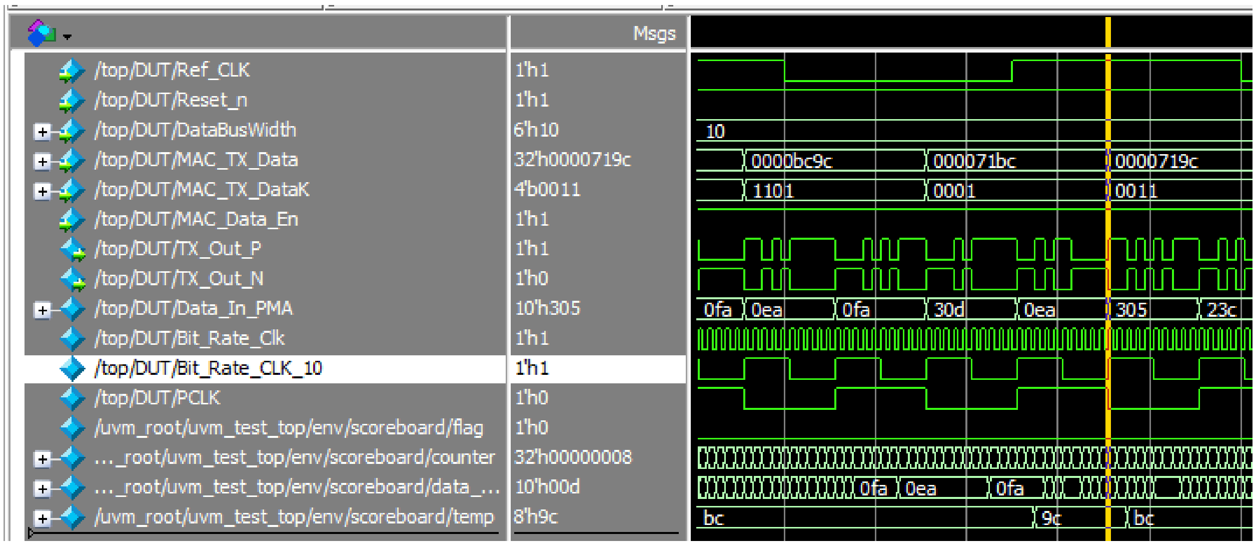
```
/top/DUT/Ref_CLK                                   1'h1
/top/DUT/Reset_n                                   1'h1
/top/DUT/DataBusWidth                              6'h08
/top/DUT/MAC_TX_Data                               32'h26c4ddbc
/top/DUT/MAC_TX_DataK                              4'b0010
/top/DUT/MAC_Data_En                               1'h1
/top/DUT/TX_Out_P                                  1'h1
/top/DUT/TX_Out_N                                  1'h0
/top/DUT/Data_In_PMA                               10'h0fa
/top/DUT/Bit_Rate_Clk                              1'h1
/top/DUT/Bit_Rate_CLK_10                           1'h1
/top/DUT/PCLK                                      1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag         1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter      32'h00000000
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h3d2
/uvm_root/uvm_test_top/env/scoreboard/temp         8'h81
```

5. **Check data 16'h719c to be sent as data with encoded value = 10'h0e2 , 10'h23c**



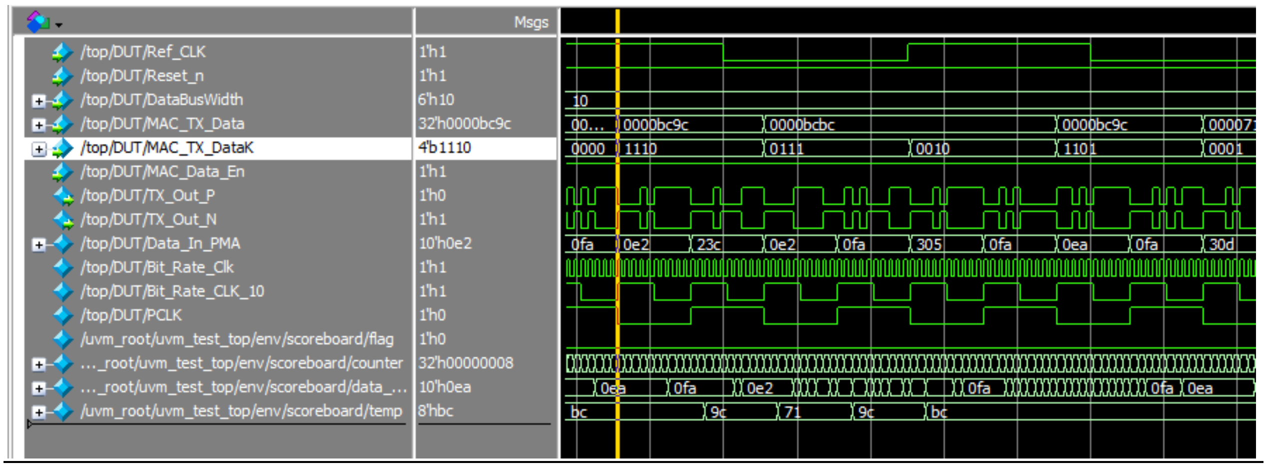
```
/top/DUT/Ref_CLK         1'h0
/top/DUT/Reset_n         1'h1
/top/DUT/DataBus...      6'h10
/top/DUT/MAC_TX_...      32'h0000bc9c
/top/DUT/MAC_TX_...      4'b1001
/top/DUT/MAC_Dat...      1'h1
/top/DUT/TX_Out_P        1'h0
/top/DUT/TX_Out_N        1'h1
/top/DUT/Data_In_...     10'h305
/top/DUT/Bit_Rate_...    1'h1
/top/DUT/Bit_Rate_...    1'h1
/top/DUT/PCLK            1'h0
/uvm_root/uvm_tes...     1'h0
/uvm_root/uvm_tes...     32'h00000008
/uvm_root/uvm_tes...     10'h22d
/uvm_root/uvm_tes...     8'h81
```

6. **Check data 16'h71bc to be sent the LSB byte as command with encoded value = 305 and the next byte which is 71 to be sent as data with encoded value = 0ea**



```
/top/DUT/Ref_CLK                               1'h1
/top/DUT/Reset_n                               1'h1
/top/DUT/DataBusWidth                          6'h10
/top/DUT/MAC_TX_Data                           32'h0000719c
/top/DUT/MAC_TX_DataK                          4'b0011
/top/DUT/MAC_Data_En                           1'h1
/top/DUT/TX_Out_P                              1'h1
/top/DUT/TX_Out_N                              1'h0
/top/DUT/Data_In_PMA                           10'h305
/top/DUT/Bit_Rate_Clk                          1'h1
/top/DUT/Bit_Rate_CLK_10                       1'h1
/top/DUT/PCLK                                  1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag     1'h0
..._root/uvm_test_top/env/scoreboard/counter   32'h00000008
..._root/uvm_test_top/env/scoreboard/data_...  10'h00d
/uvm_root/uvm_test_top/env/scoreboard/temp     8'h9c
```

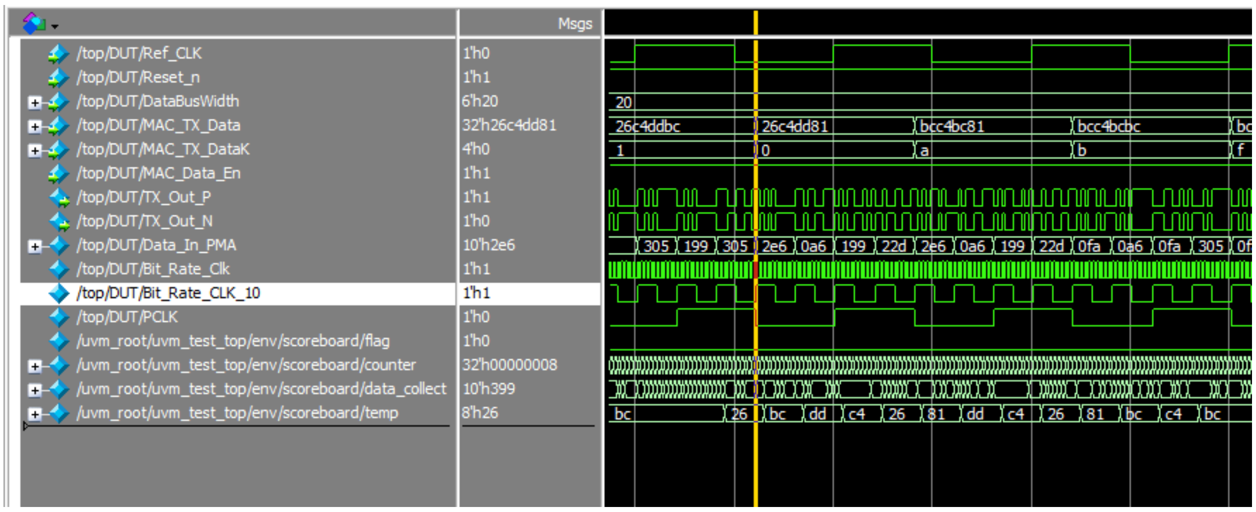
7. **Check data 16'hbc9c and 16'hbcbc patterns to be sent with constraints for command or data as following :**
    a. **1st pattern which is 16'hbc9c sent the LSB byte as data and the 2nd one as command**
    b. **2nd pattern which is 16'hbcbc sent all two bytes as command**



	Msgs
/top/DUT/Ref_CLK	1'h1
/top/DUT/Reset_n	1'h1
/top/DUT/DataBusWidth	6'h10
/top/DUT/MAC_TX_Data	32'h0000bc9c
/top/DUT/MAC_TX_DataK	4'b1110
/top/DUT/MAC_Data_En	1'h1
/top/DUT/TX_Out_P	1'h0
/top/DUT/TX_Out_N	1'h1
/top/DUT/Data_In_PMA	10'h0e2
/top/DUT/Bit_Rate_Clk	1'h1
/top/DUT/Bit_Rate_CLK_10	1'h1
/top/DUT/PCLK	1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag	1'h0
…_root/uvm_test_top/env/scoreboard/counter	32'h00000008
…_root/uvm_test_top/env/scoreboard/data_…	10'h0ea
/uvm_root/uvm_test_top/env/scoreboard/temp	8'hbc

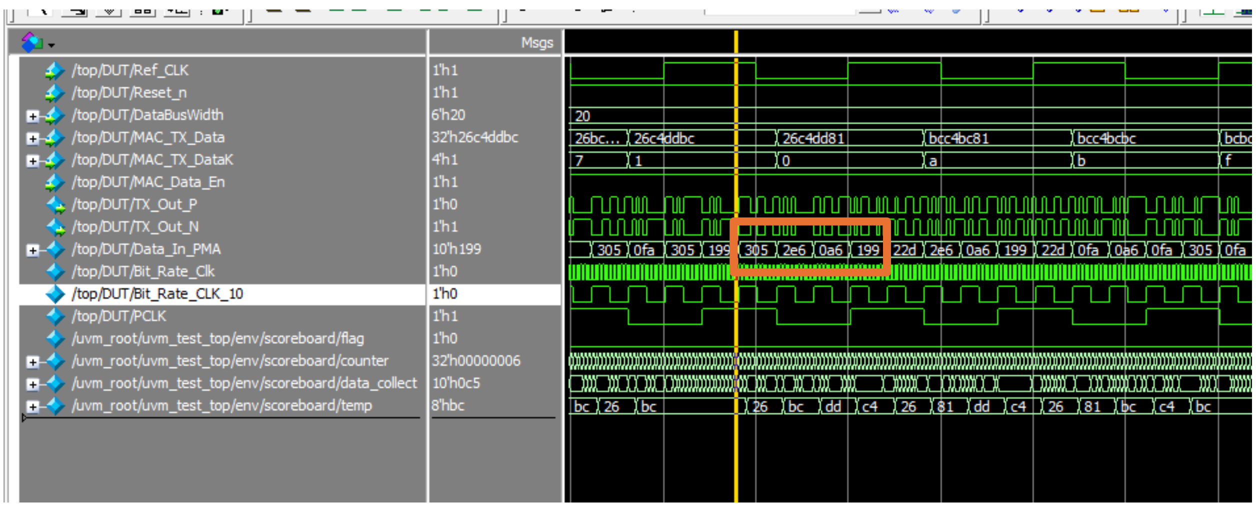
Width 32

8. **Check data 32'h26c4dd81 to be sent all as data with encoded value = 22d – 2e6-0a6-199 alternatively positive & negative RD**



	Msgs
/top/DUT/Ref_CLK	1'h0
/top/DUT/Reset_n	1'h1
/top/DUT/DataBusWidth	6'h20
/top/DUT/MAC_TX_Data	32'h26c4dd81
/top/DUT/MAC_TX_DataK	4'h0
/top/DUT/MAC_Data_En	1'h1
/top/DUT/TX_Out_P	1'h1
/top/DUT/TX_Out_N	1'h0
/top/DUT/Data_In_PMA	10'h2e6
/top/DUT/Bit_Rate_Clk	1'h1
/top/DUT/Bit_Rate_CLK_10	1'h1
/top/DUT/PCLK	1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag	1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter	32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect	10'h399
/uvm_root/uvm_test_top/env/scoreboard/temp	8'h26

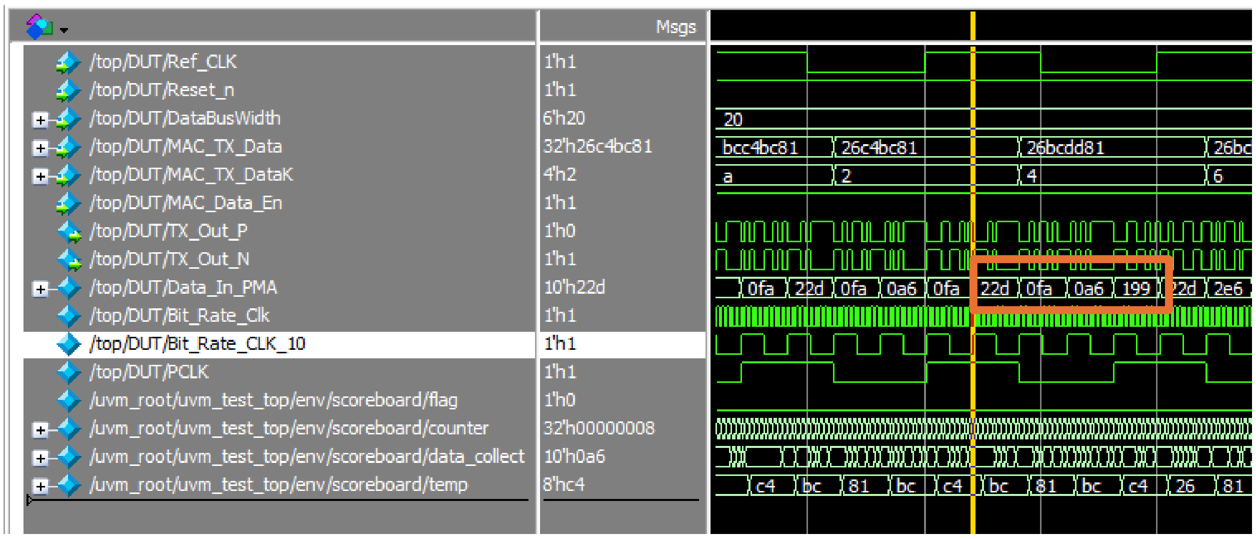
9. **Check data 32'h26c4ddbc to be sent the LSB byte as command and the other bytes are sent as actual Data**



```
/top/DUT/Ref_CLK                                  1'h1
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'h26c4ddbc
/top/DUT/MAC_TX_DataK                             4'h1
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h0
/top/DUT/TX_Out_N                                 1'h1
/top/DUT/Data_In_PMA                              10'h199
/top/DUT/Bit_Rate_Clk                             1'h0
/top/DUT/Bit_Rate_CLK_10                          1'h0
/top/DUT/PCLK                                     1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000006
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h0c5
/uvm_root/uvm_test_top/env/scoreboard/temp        8'hbc
```

Msgs
20
26bc... 26c4ddbc 26c4dd81 bcc4bc81 bcc4bcbc bcbc
7 1 0 a b f
305 0fa 305 199 305 2e6 0a6 199 22d 2e6 0a6 199 22d 0fa 0a6 0fa 305 0fa
bc 26 bc 26 bc dd c4 26 81 dd c4 26 81 bc c4 bc

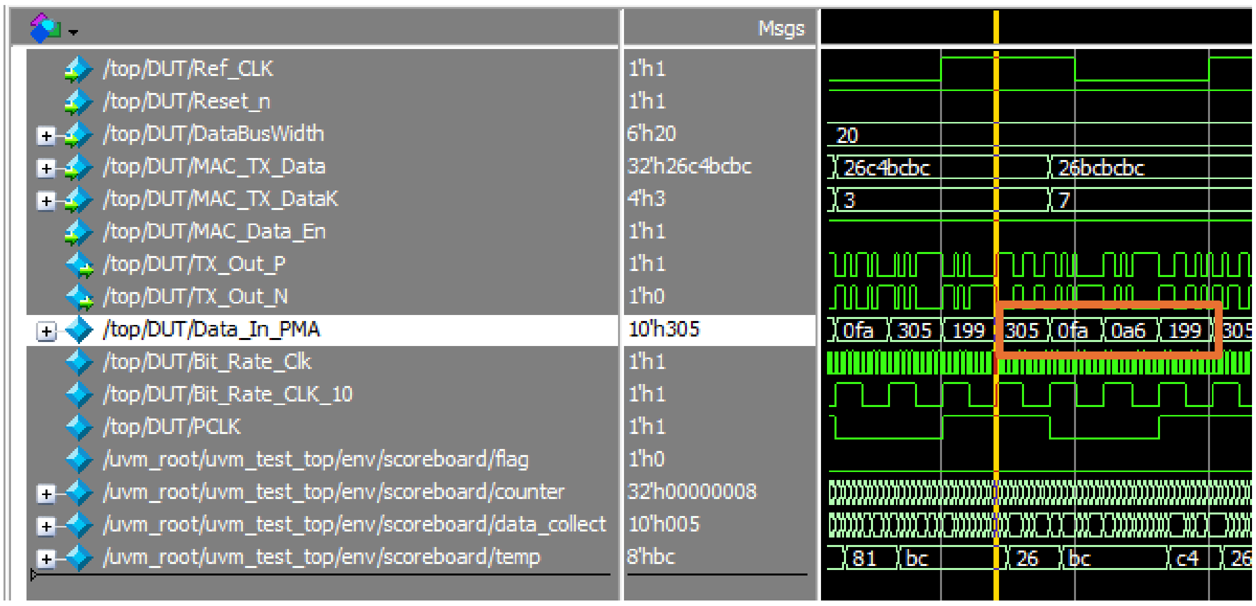
10. Check data 32'h26c4bc81 to be all sent as Data except the 2nd byte to be sent as command :



```
/top/DUT/Ref_CLK                                  1'h1
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'h26c4bc81
/top/DUT/MAC_TX_DataK                             4'h2
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h0
/top/DUT/TX_Out_N                                 1'h1
/top/DUT/Data_In_PMA                              10'h22d
/top/DUT/Bit_Rate_Clk                             1'h1
/top/DUT/Bit_Rate_CLK_10                          1'h1
/top/DUT/PCLK                                     1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h0a6
/uvm_root/uvm_test_top/env/scoreboard/temp        8'hc4
```

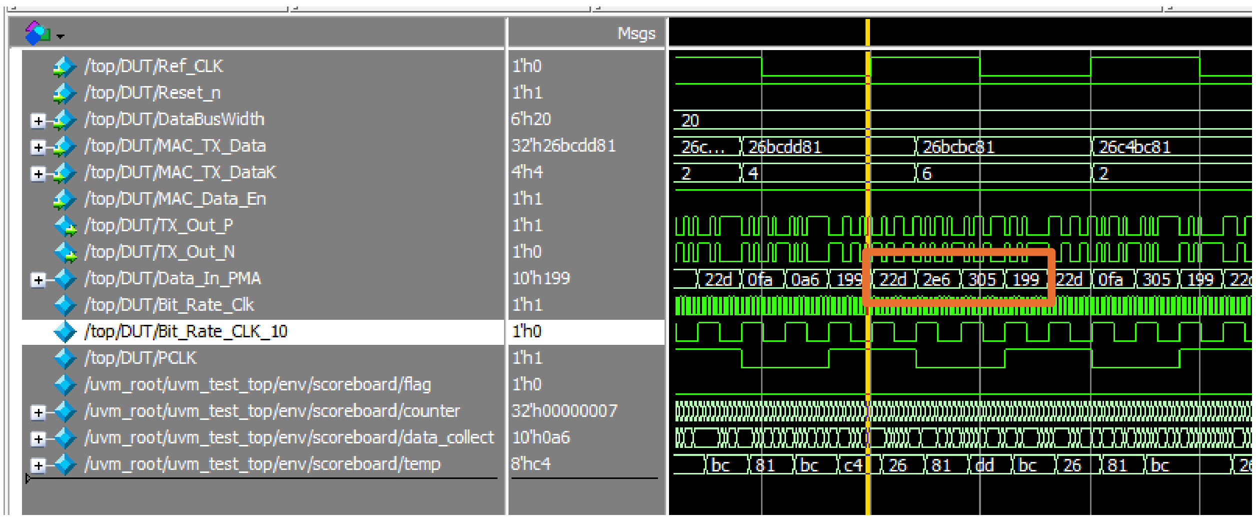
Msgs
20
bcc4bc81 26c4bc81 26bcdd81 26bc
a 2 4 6
0fa 22d 0fa 0a6 0fa 22d 0fa 0a6 199 22d 2e6
c4 bc 81 bc c4 bc 81 bc c4 26 81

11. Check data 32'h26c4bcbc to be sent as data except the 1st & 2nd byte to be sent as command :



```
/top/DUT/Ref_CLK                                  1'h1
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'h26c4bcbc
/top/DUT/MAC_TX_DataK                             4'h3
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h1
/top/DUT/TX_Out_N                                 1'h0
/top/DUT/Data_In_PMA                              10'h305
/top/DUT/Bit_Rate_Clk                             1'h1
/top/DUT/Bit_Rate_CLK_10                          1'h1
/top/DUT/PCLK                                     1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h005
/uvm_root/uvm_test_top/env/scoreboard/temp        8'hbc
```

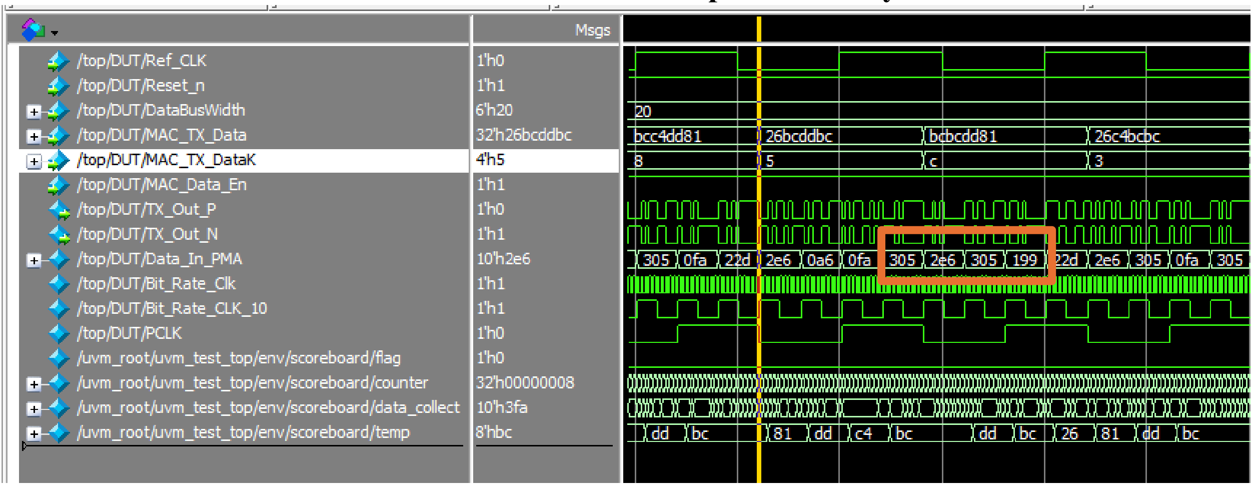
12. Check data 32'h26bcdd81 to be sent all as data except the 3rd byte :



```
/top/DUT/Ref_CLK                               1'h0
/top/DUT/Reset_n                               1'h1
/top/DUT/DataBusWidth                          6'h20
/top/DUT/MAC_TX_Data                           32'h26bcdd81
/top/DUT/MAC_TX_DataK                          4'h4
/top/DUT/MAC_Data_En                           1'h1
/top/DUT/TX_Out_P                              1'h1
/top/DUT/TX_Out_N                              1'h0
/top/DUT/Data_In_PMA                           10'h199
/top/DUT/Bit_Rate_Clk                          1'h1
/top/DUT/Bit_Rate_CLK_10                       1'h0
/top/DUT/PCLK                                  1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag          1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter       32'h00000007
/uvm_root/uvm_test_top/env/scoreboard/data_collect  10'h0a6
/uvm_root/uvm_test_top/env/scoreboard/temp          8'hc4
```

20
26c... 26bcdd81 26bcbc81 26c4bc81
2 4 6 2
22d 0fa 0a6 199 22d 2e6 305 199 22d 0fa 305 199 22
bc 81 bc c4 26 81 dd bc 26 81 bc 2

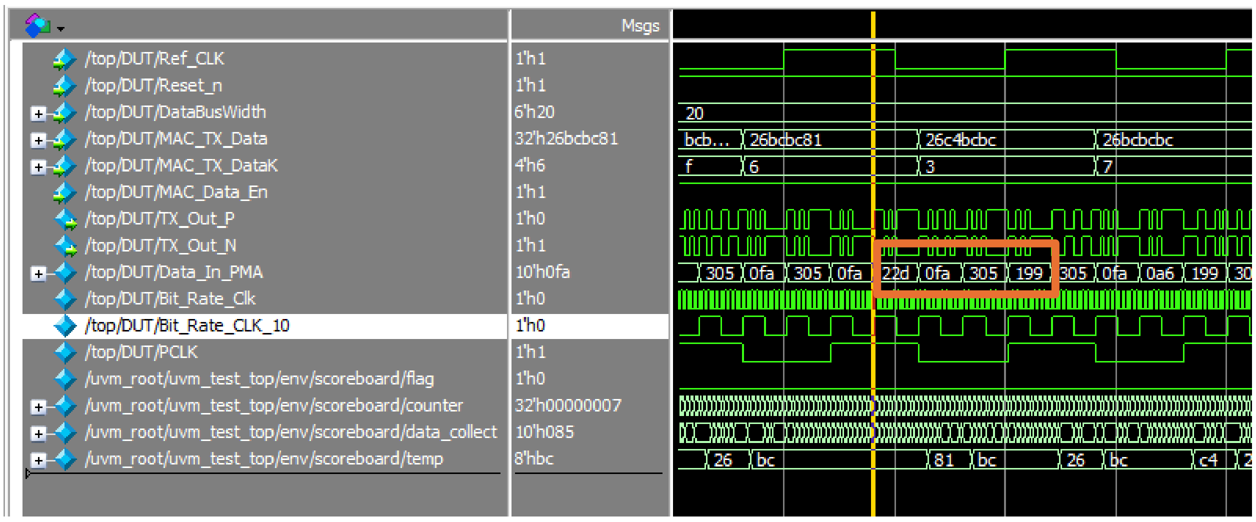
13. **Check data 32'h26bcddbc to be sent all as data except 1st & 3rd bytes to be sent as command :**



```
/top/DUT/Ref_CLK                               1'h0
/top/DUT/Reset_n                               1'h1
/top/DUT/DataBusWidth                          6'h20
/top/DUT/MAC_TX_Data                           32'h26bcddbc
/top/DUT/MAC_TX_DataK                          4'h5
/top/DUT/MAC_Data_En                           1'h1
/top/DUT/TX_Out_P                              1'h0
/top/DUT/TX_Out_N                              1'h1
/top/DUT/Data_In_PMA                           10'h2e6
/top/DUT/Bit_Rate_Clk                          1'h1
/top/DUT/Bit_Rate_CLK_10                       1'h1
/top/DUT/PCLK                                  1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag          1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter       32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect  10'h3fa
/uvm_root/uvm_test_top/env/scoreboard/temp          8'hbc
```

20
bcc4dd81 26bcddbc bcbcdd81 26c4bcbc
8 5 c 3
305 0fa 22d 2e6 0a6 0fa 305 2e6 305 199 22d 2e6 305 0fa 305
dd bc 81 dd c4 bc dd bc 26 81 dd bc

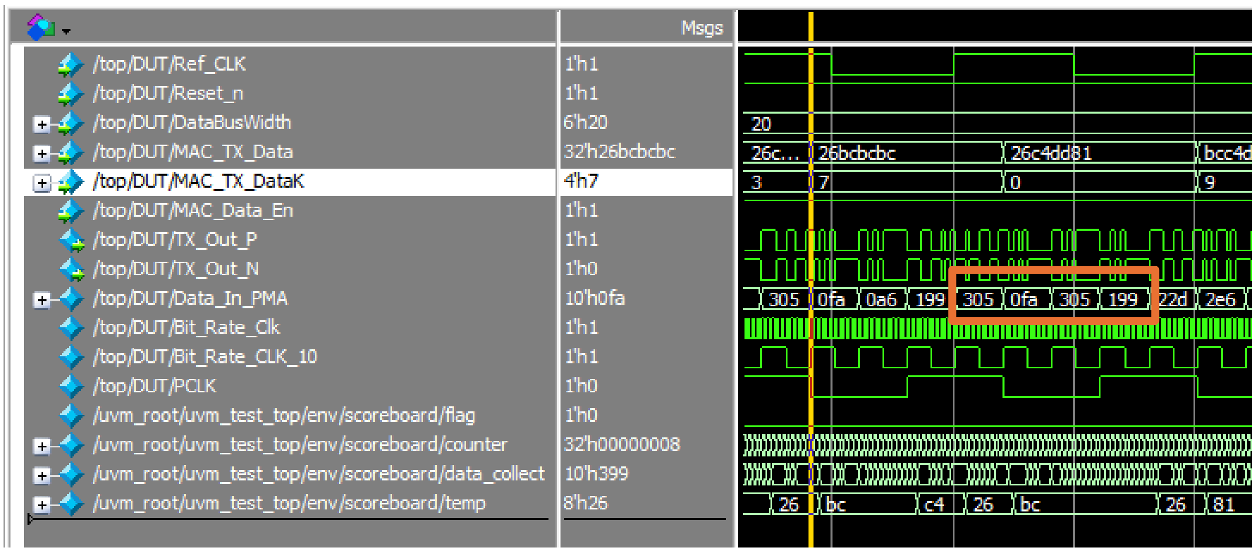
14. **Check data 32'h26bcbc81 to be sent as data except the 2nd and 3rd bytes are sent as command :**



```
/top/DUT/Ref_CLK                               1'h1
/top/DUT/Reset_n                               1'h1
/top/DUT/DataBusWidth                          6'h20
/top/DUT/MAC_TX_Data                           32'h26bcbc81
/top/DUT/MAC_TX_DataK                          4'h6
/top/DUT/MAC_Data_En                           1'h1
/top/DUT/TX_Out_P                              1'h0
/top/DUT/TX_Out_N                              1'h1
/top/DUT/Data_In_PMA                           10'h0fa
/top/DUT/Bit_Rate_Clk                          1'h0
/top/DUT/Bit_Rate_CLK_10                       1'h0
/top/DUT/PCLK                                  1'h1
/uvm_root/uvm_test_top/env/scoreboard/flag          1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter       32'h00000007
/uvm_root/uvm_test_top/env/scoreboard/data_collect  10'h085
/uvm_root/uvm_test_top/env/scoreboard/temp          8'hbc
```

20
bcb... 26bcbc81 26c4bcbc 26bcbcbc
f 6 3 7
305 0fa 305 0fa 22d 0fa 305 199 305 0fa 0a6 199 30
26 bc 81 bc 26 bc c4 2

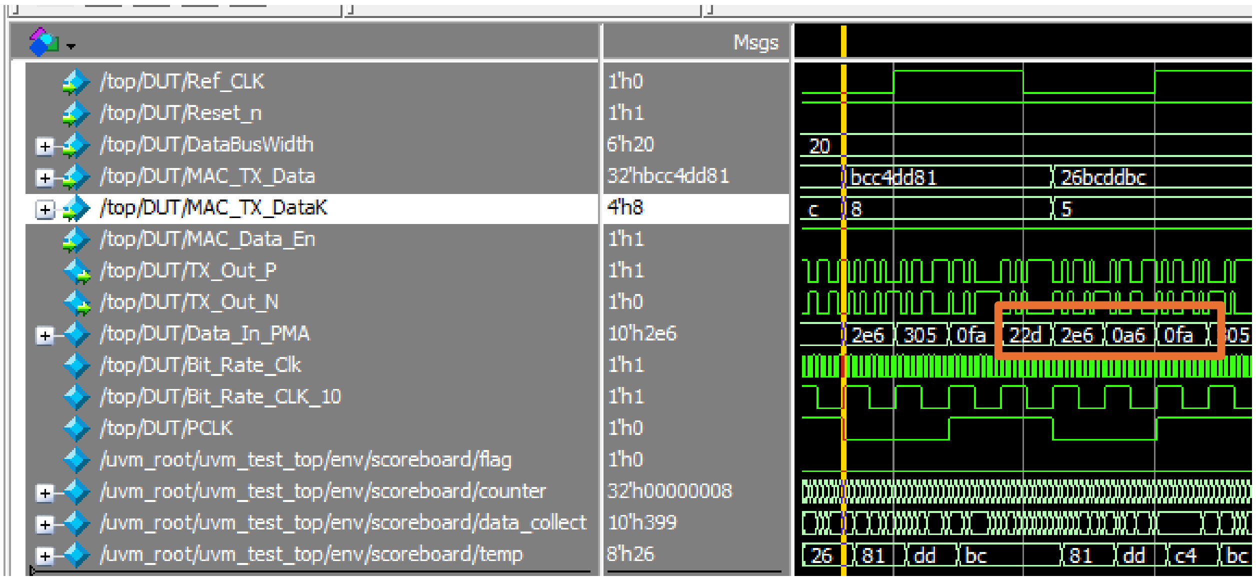
15. **Check data 32'h26bcbcbc to be all sent as command except the MSB byte is sent as Data:**



```
Msgs
/top/DUT/Ref_CLK                                  1'h1
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'h26bcbcbc
/top/DUT/MAC_TX_DataK                             4'h7
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h1
/top/DUT/TX_Out_N                                 1'h0
/top/DUT/Data_In_PMA                              10'h0fa
/top/DUT/Bit_Rate_Clk                             1'h1
/top/DUT/Bit_Rate_CLK_10                          1'h1
/top/DUT/PCLK                                     1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h399
/uvm_root/uvm_test_top/env/scoreboard/temp        8'h26
```

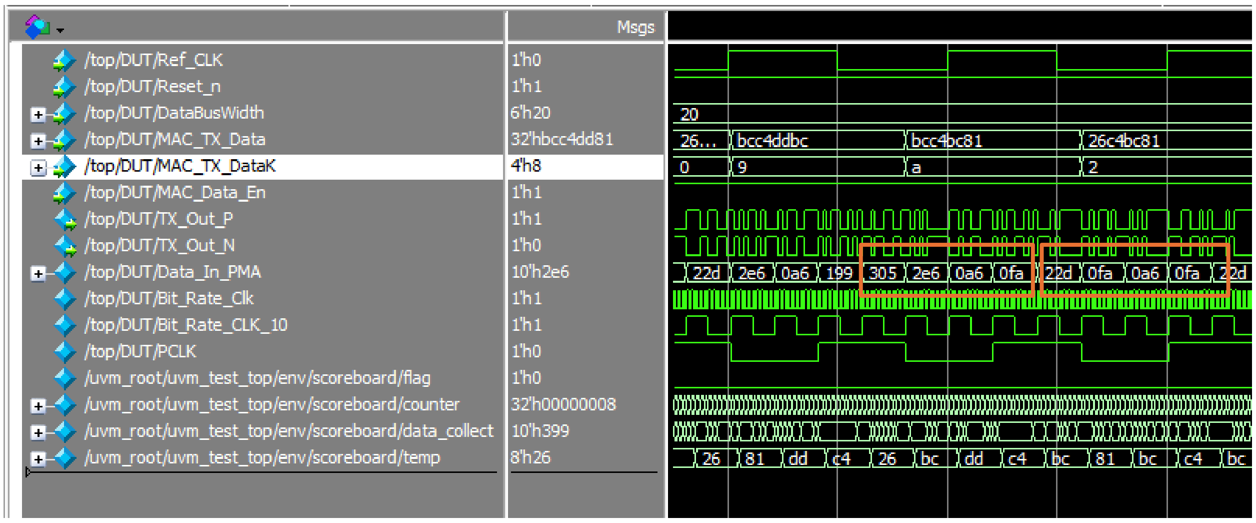
20
26c... 26bcbcbc 26c4dd81 bcc4d
3 7 0 9
305 0fa 0a6 199 305 0fa 305 199 22d 2e6
26 bc c4 26 bc 26 81

16. Check data 32’hbcc4dd81 to be sent as data except MSB byte to be sent as command :



```
Msgs
/top/DUT/Ref_CLK                                  1'h0
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'hbcc4dd81
/top/DUT/MAC_TX_DataK                             4'h8
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h1
/top/DUT/TX_Out_N                                 1'h0
/top/DUT/Data_In_PMA                              10'h2e6
/top/DUT/Bit_Rate_Clk                             1'h1
/top/DUT/Bit_Rate_CLK_10                          1'h1
/top/DUT/PCLK                                     1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h399
/uvm_root/uvm_test_top/env/scoreboard/temp        8'h26
```

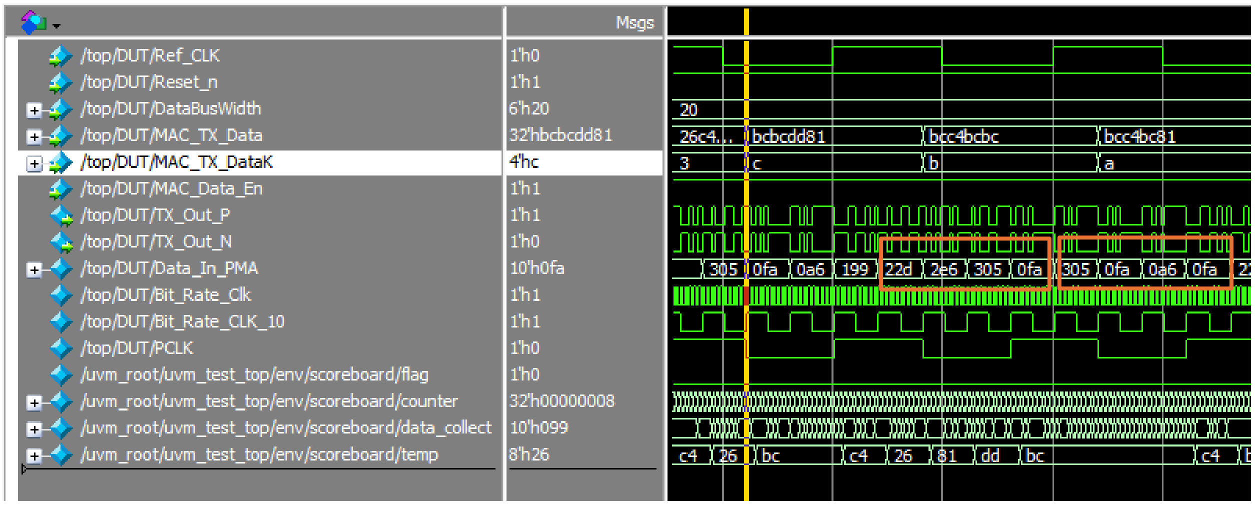
17. Check data 32’hbcc4ddbc to be sent as data except the LSB,MSB bytes are sent as command and data 32’hbcc4bc81 to be sent as data except the 2nd & 4th bytes are sent as command :



```
Msgs
/top/DUT/Ref_CLK                                  1'h0
/top/DUT/Reset_n                                  1'h1
/top/DUT/DataBusWidth                             6'h20
/top/DUT/MAC_TX_Data                              32'hbcc4dd81
/top/DUT/MAC_TX_DataK                             4'h8
/top/DUT/MAC_Data_En                              1'h1
/top/DUT/TX_Out_P                                 1'h1
/top/DUT/TX_Out_N                                 1'h0
/top/DUT/Data_In_PMA                              10'h2e6
/top/DUT/Bit_Rate_Clk                             1'h1
/top/DUT/Bit_Rate_CLK_10                          1'h1
/top/DUT/PCLK                                     1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag        1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter     32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect 10'h399
/uvm_root/uvm_test_top/env/scoreboard/temp        8'h26
```

20
26... bcc4ddbc bcc4bc81 26c4bc81
0 9 a 2
22d 2e6 0a6 199 305 2e6 0a6 0fa 22d 0fa 0a6 0fa 22d
26 81 dd c4 26 bc dd c4 bc 81 bc c4 bc

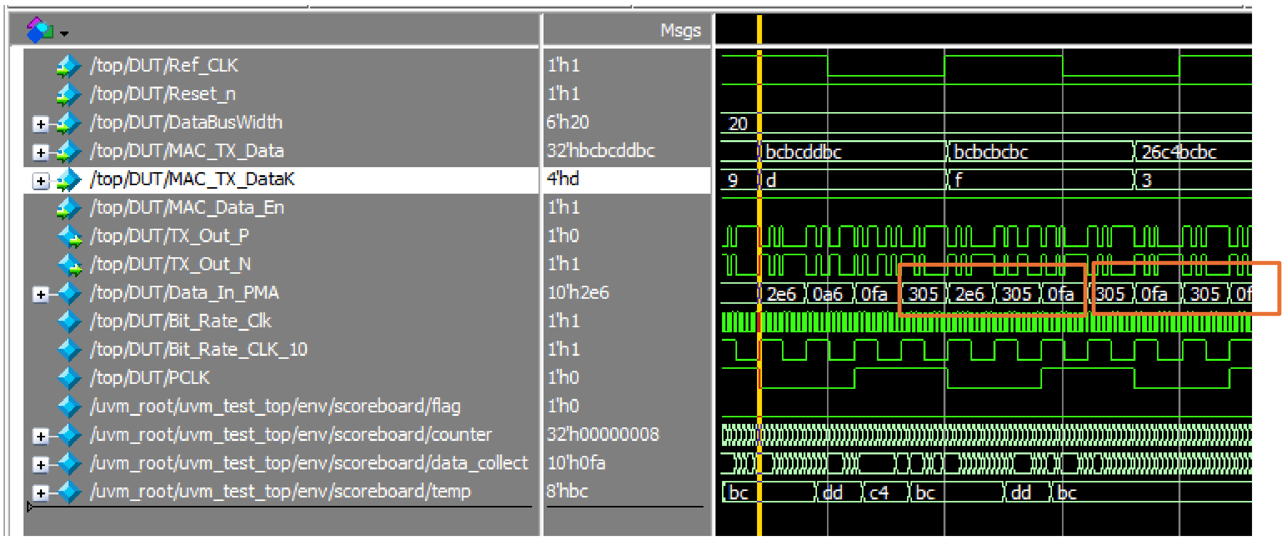
18. Check data 32’hbcc4bcbc to be sent as command except the 3rd byte is sent as data and data 32’hbcbcdd81 to be sent as MSB 2-bytes as command and LSB 2-bytes are sent as Data :



	Msgs
/top/DUT/Ref_CLK	1'h0
/top/DUT/Reset_n	1'h1
/top/DUT/DataBusWidth	6'h20
/top/DUT/MAC_TX_Data	32'hbcbcdd81
/top/DUT/MAC_TX_DataK	4'hc
/top/DUT/MAC_Data_En	1'h1
/top/DUT/TX_Out_P	1'h1
/top/DUT/TX_Out_N	1'h0
/top/DUT/Data_In_PMA	10'h0fa
/top/DUT/Bit_Rate_Clk	1'h1
/top/DUT/Bit_Rate_CLK_10	1'h1
/top/DUT/PCLK	1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag	1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter	32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect	10'h099
/uvm_root/uvm_test_top/env/scoreboard/temp	8'h26

20
26c4... bcbcdd81 bcc4bcbc bcc4bc81
3 c b a
305 0fa 0a6 199 22d 2e6 305 0fa 305 0fa 0a6 0fa
c4 26 bc c4 26 81 dd bc c4

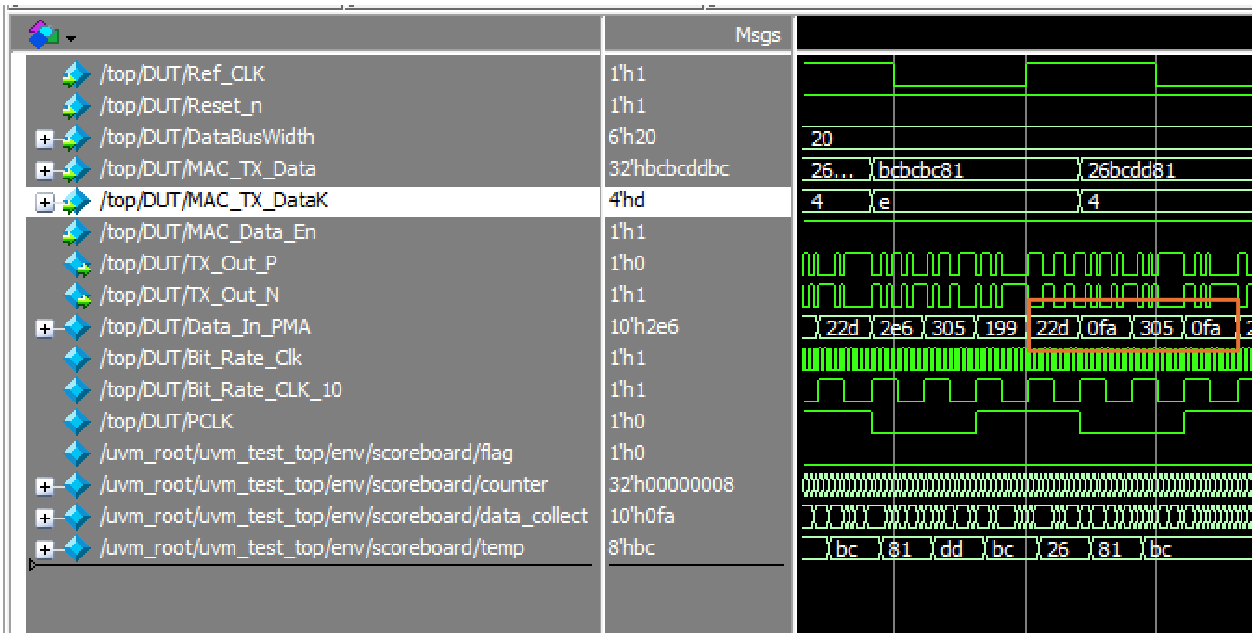
19. Check data 32’hbcbcddbc to be sent as command except 2nd byte is sent as data and data 32’hbcbcbcbc to be sent all as command :



	Msgs
/top/DUT/Ref_CLK	1'h1
/top/DUT/Reset_n	1'h1
/top/DUT/DataBusWidth	6'h20
/top/DUT/MAC_TX_Data	32'hbcbcddbc
/top/DUT/MAC_TX_DataK	4'hd
/top/DUT/MAC_Data_En	1'h1
/top/DUT/TX_Out_P	1'h0
/top/DUT/TX_Out_N	1'h1
/top/DUT/Data_In_PMA	10'h2e6
/top/DUT/Bit_Rate_Clk	1'h1
/top/DUT/Bit_Rate_CLK_10	1'h1
/top/DUT/PCLK	1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag	1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter	32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect	10'h0fa
/uvm_root/uvm_test_top/env/scoreboard/temp	8'hbc

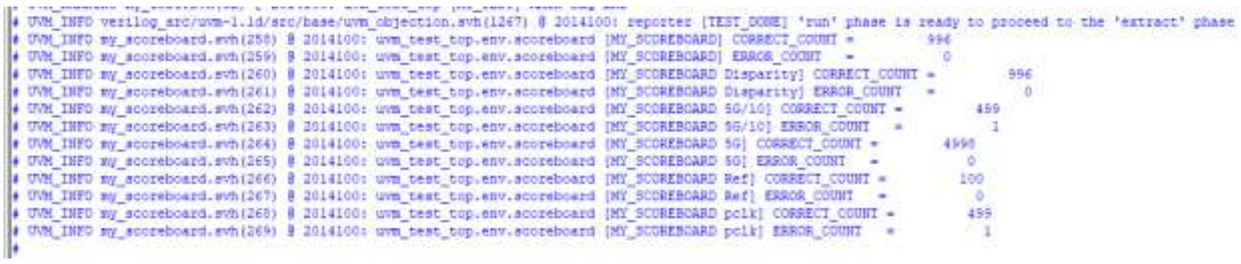
20
bcbcddbc bcbcbcbc 26c4bcbc
9 d f 3
2e6 0a6 0fa 305 2e6 305 0fa 305 0fa 305
bc dd c4 bc dd bc

20. Check data 32’hbcbcbc81 to be sent as command except the LSB byte is sent as data :



	Msgs
/top/DUT/Ref_CLK	1'h1
/top/DUT/Reset_n	1'h1
/top/DUT/DataBusWidth	6'h20
/top/DUT/MAC_TX_Data	32'hbcbcddbc
/top/DUT/MAC_TX_DataK	4'hd
/top/DUT/MAC_Data_En	1'h1
/top/DUT/TX_Out_P	1'h0
/top/DUT/TX_Out_N	1'h1
/top/DUT/Data_In_PMA	10'h2e6
/top/DUT/Bit_Rate_Clk	1'h1
/top/DUT/Bit_Rate_CLK_10	1'h1
/top/DUT/PCLK	1'h0
/uvm_root/uvm_test_top/env/scoreboard/flag	1'h0
/uvm_root/uvm_test_top/env/scoreboard/counter	32'h00000008
/uvm_root/uvm_test_top/env/scoreboard/data_collect	10'h0fa
/uvm_root/uvm_test_top/env/scoreboard/temp	8'hbc

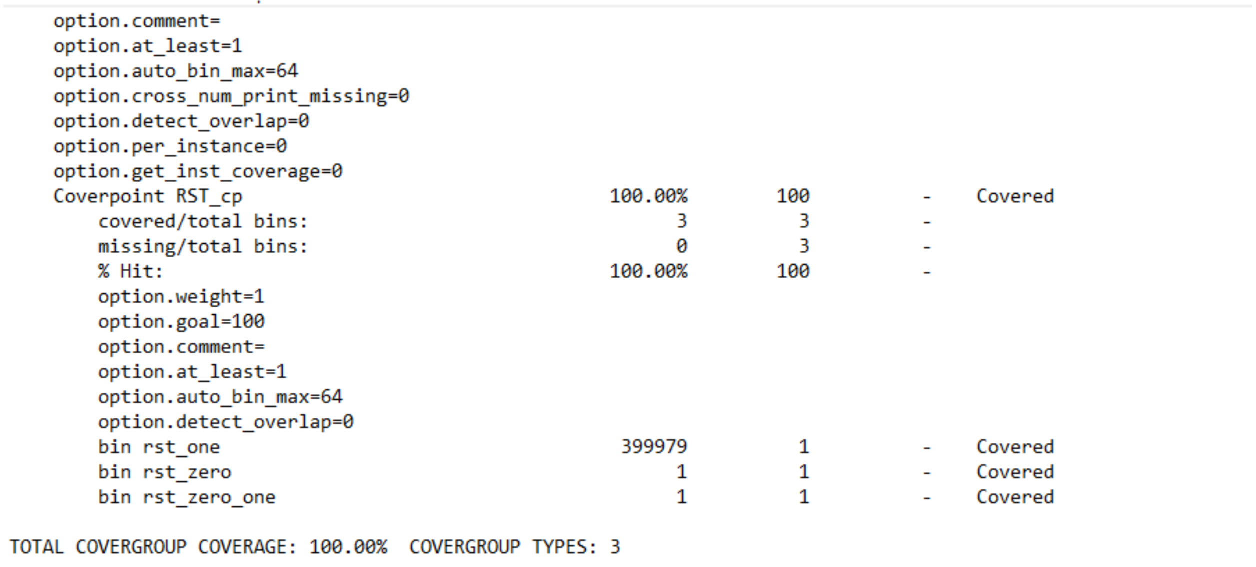
UVM_Report :



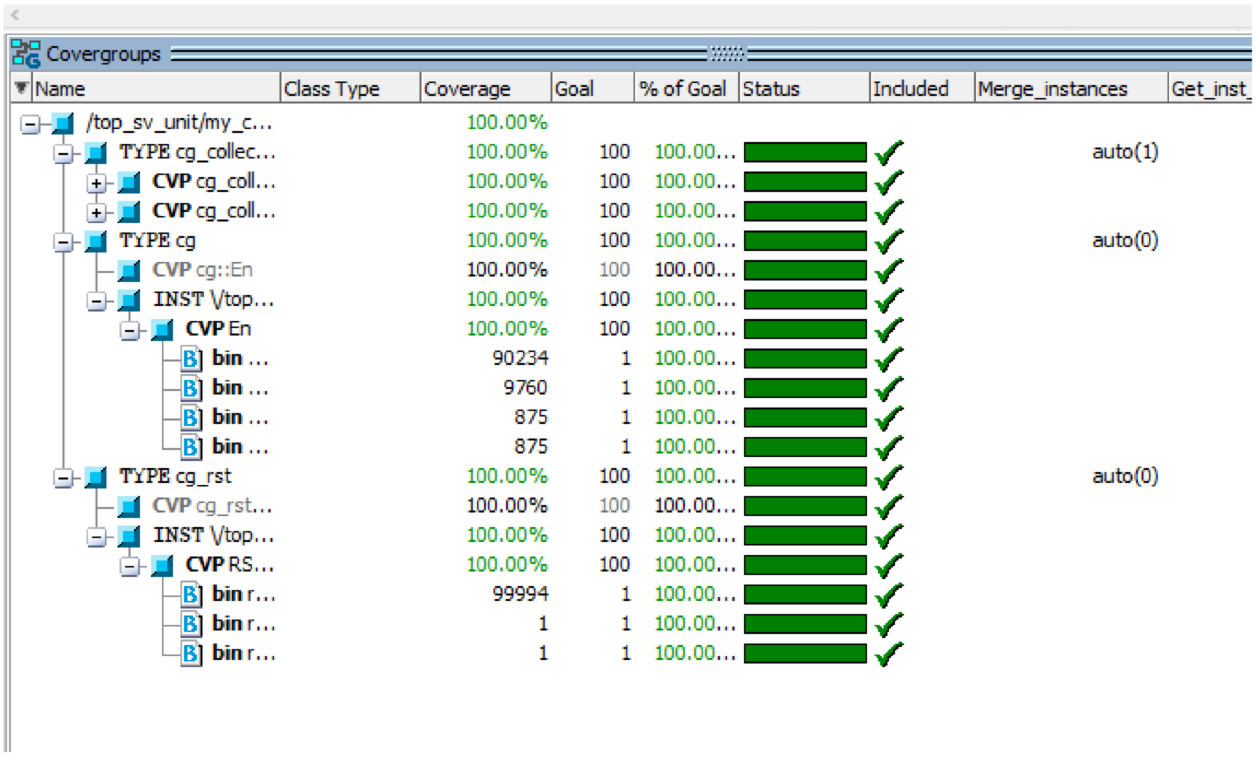
```
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 2014100: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
# UVM_INFO my_scoreboard.svh(258) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD] CORRECT_COUNT =          996
# UVM_INFO my_scoreboard.svh(259) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD] ERROR_COUNT   =            0
# UVM_INFO my_scoreboard.svh(260) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD Disparity] CORRECT_COUNT =          996
# UVM_INFO my_scoreboard.svh(261) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD Disparity] ERROR_COUNT   =            0
# UVM_INFO my_scoreboard.svh(262) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD 5G/10] CORRECT_COUNT =          499
# UVM_INFO my_scoreboard.svh(263) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD 5G/10] ERROR_COUNT   =            1
# UVM_INFO my_scoreboard.svh(264) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD 5G] CORRECT_COUNT =         4998
# UVM_INFO my_scoreboard.svh(265) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD 5G] ERROR_COUNT   =            0
# UVM_INFO my_scoreboard.svh(266) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD Ref] CORRECT_COUNT =          100
# UVM_INFO my_scoreboard.svh(267) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD Ref] ERROR_COUNT   =            0
# UVM_INFO my_scoreboard.svh(268) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD pclk] CORRECT_COUNT =          499
# UVM_INFO my_scoreboard.svh(269) @ 2014100: uvm_test_top.env.scoreboard [MY_SCOREBOARD pclk] ERROR_COUNT   =            1
#
```

Coverage Reports :

1. Functional Coverage :



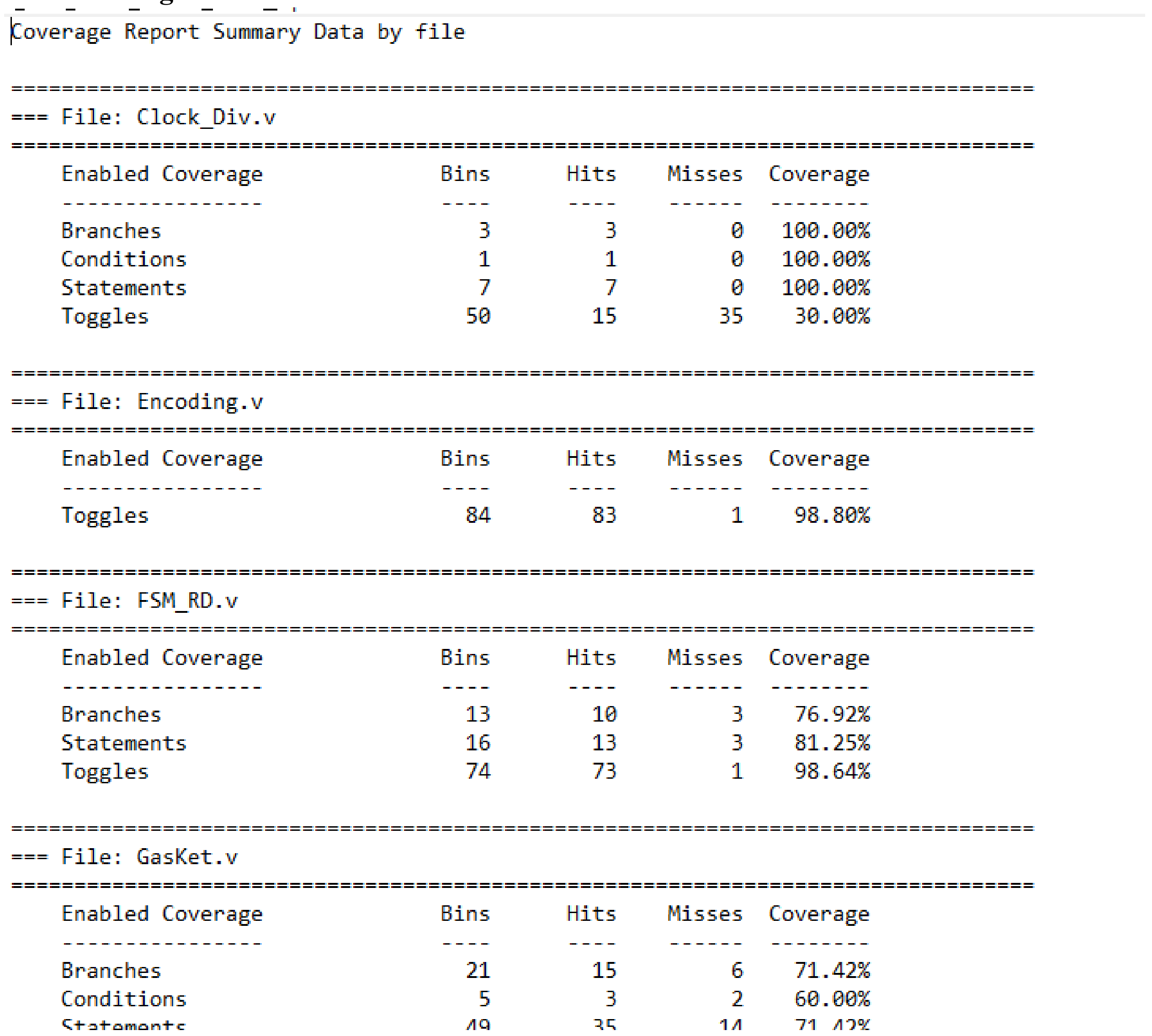
```
        option.comment=
        option.at_least=1
        option.auto_bin_max=64
        option.cross_num_print_missing=0
        option.detect_overlap=0
        option.per_instance=0
        option.get_inst_coverage=0
        Coverpoint RST_cp                          100.00%        100          -    Covered
            covered/total bins:                          3          3          -
            missing/total bins:                          0          3          -
            % Hit:                                 100.00%        100          -
            option.weight=1
            option.goal=100
            option.comment=
            option.at_least=1
            option.auto_bin_max=64
            option.detect_overlap=0
            bin rst_one                             399979          1          -    Covered
            bin rst_zero                                 1          1          -    Covered
            bin rst_zero_one                             1          1          -    Covered

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 3
```



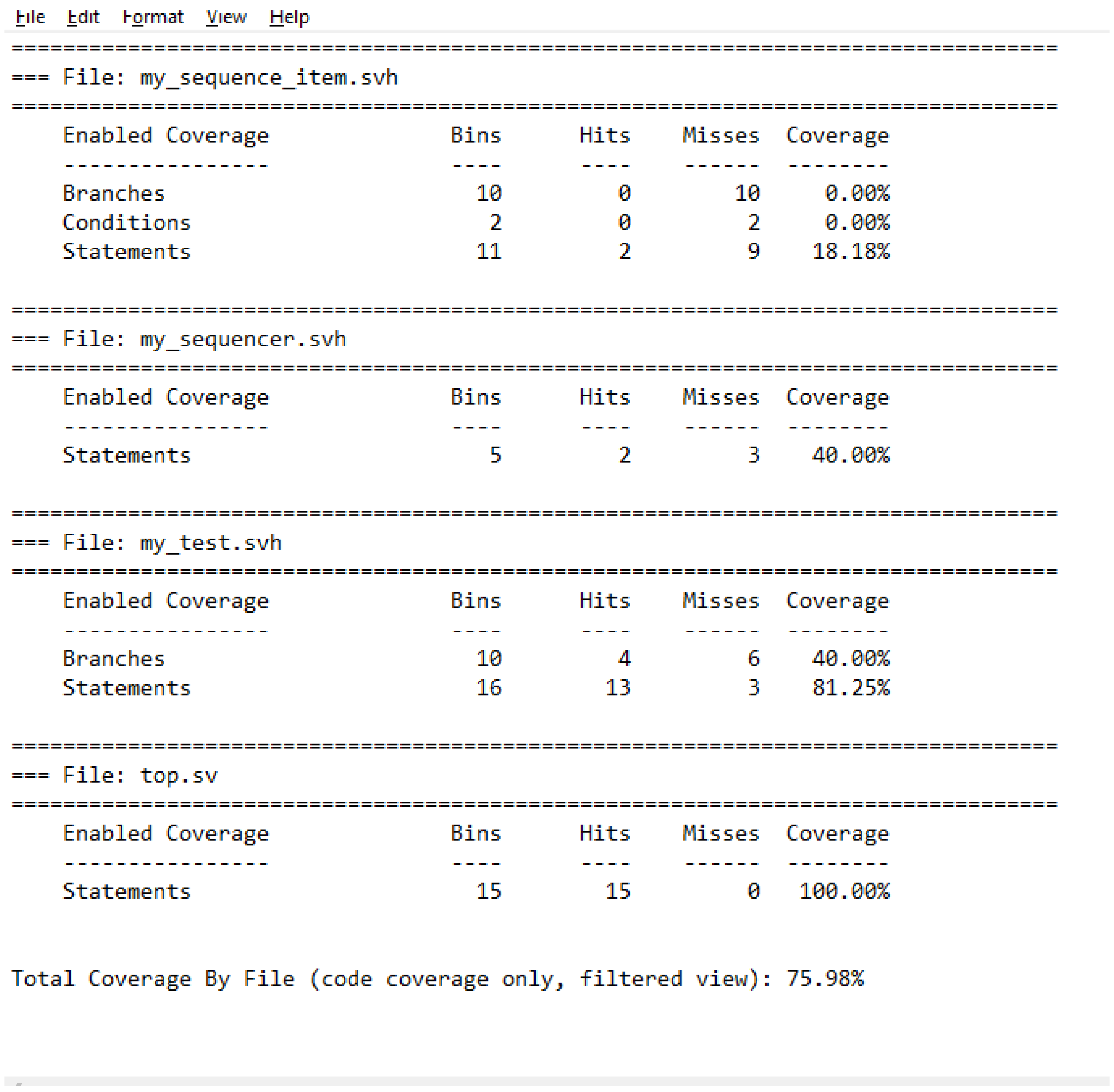
Covergroups

Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_
/top_sv_unit/my_c...		100.00%						
TYPE cg_collec...		100.00%	100	100.00...	🟩	✓	auto(1)	
CVP cg_coll...		100.00%	100	100.00...	🟩	✓		
CVP cg_coll...		100.00%	100	100.00...	🟩	✓		
TYPE cg		100.00%	100	100.00...	🟩	✓	auto(0)	
CVP cg::En		100.00%	100	100.00...	🟩	✓		
INST \/top...		100.00%	100	100.00...	🟩	✓		
CVP En		100.00%	100	100.00...	🟩	✓		
bin ...		90234	1	100.00...	🟩	✓		
bin ...		9760	1	100.00...	🟩	✓		
bin ...		875	1	100.00...	🟩	✓		
bin ...		875	1	100.00...	🟩	✓		
TYPE cg_rst		100.00%	100	100.00...	🟩	✓	auto(0)	
CVP cg_rst...		100.00%	100	100.00...	🟩	✓		
INST \/top...		100.00%	100	100.00...	🟩	✓		
CVP RS...		100.00%	100	100.00...	🟩	✓		
bin r...		99994	1	100.00...	🟩	✓		
bin r...		1	1	100.00...	🟩	✓		
bin r...		1	1	100.00...	🟩	✓		

2. Code coverage :



```
Coverage Report Summary Data by file

=================================================================================
=== File: Clock_Div.v
=================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                         3         3         0   100.00%
    Conditions                       1         1         0   100.00%
    Statements                       7         7         0   100.00%
    Toggles                         50        15        35    30.00%

=================================================================================
=== File: Encoding.v
=================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         84        83         1    98.80%

=================================================================================
=== File: FSM_RD.v
=================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        13        10         3    76.92%
    Statements                      16        13         3    81.25%
    Toggles                         74        73         1    98.64%

=================================================================================
=== File: GasKet.v
=================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        21        15         6    71.42%
    Conditions                       5         3         2    60.00%
    Statements                      49        35        14    71.42%
```



```
================================================================================
=== File: my_sequence_item.svh
================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        10         0        10     0.00%
    Conditions                       2         0         2     0.00%
    Statements                      11         2         9    18.18%

================================================================================
=== File: my_sequencer.svh
================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                       5         2         3    40.00%

================================================================================
=== File: my_test.svh
================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        10         4         6    40.00%
    Statements                      16        13         3    81.25%

================================================================================
=== File: top.sv
================================================================================
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      15        15         0   100.00%


Total Coverage By File (code coverage only, filtered view): 75.98%
```

References

1) Universal Serial Bus 3.1 Specification
https://drive.google.com/file/d/1ZBBd8OX6Sc8p1ben5k2sk_4LWaJ1RjEU/view

2) PHY Interface For the PCI Express* and USB 3.0 Architectures
https://drive.google.com/file/d/12g_6AZ4udsnUtlb2-_bJfPL_TDmbZ_JZ/view