# 25GEMAC/PCS+RS-FEC IP Core

March 31, 2023 Product Specification Rev1.0

.

## Design Gateway Co.,Ltd

E-mail: ip-sales@design-gateway.com
URL: design-gateway.com

## Core Facts

| Provided with Core | |
|---|---|
| Documentation | User Guide, Design Guide |
| Design File Formats | Encrypted file |
| Instantiation Templates | VHDL |
| Reference Designs & Application Notes | Vivado Project,<br>See Reference Design Manual |
| Additional Items | Demo on KCU116 and ZCU111 |
| **Support** | |
| Support Provided by Design Gateway Co., Ltd. | |

## Features

- 25 Gbps Ethernet MAC and PCS with RS-FEC conforming to IEEE 802.3 by and 25G Ethernet Consortium standard
- User interface: 64-bit AXI4 stream, compatible with DG TOE25G IP and UDP25G IP
- Single clock domain for user interface, 390.625 MHz output from the Xilinx transceiver
- Xilinx transceiver interface: 64-bit data width with enabling Asynchronous Gearbox for 64B/66B
- Low latency time: 473.60 ns (average time), resulting from the round-trip time from Tx to Rx user interface using SFP28 Loopback module
- Optimized resource utilization
- Supports transmit packet length: 1 – 9014 bytes with zero-padding capability
- Supports Frame Check Sequence insertion for transmitted frames
- Supports Frame Check Sequence error detection and marking for received frames
- Supports 25GBASE-R PCS sublayer operating at 25.78125 Gbps
- Implements RS-FEC core using RS(528, 514, 10) specified in clause 108, with error indication bypass capability
- Customized service for the following features
  - Latency time improvement
  - Ethernet flow control and congestion management
  - Fully or other bypass RS-FEC capabilities

**Table 1: Example Implementation Statistics for Ultrascale+ device**

| Family | Example Device | Fmax (MHz) | CLB Regs | CLB LUTs | CLB[1] | IOB | BRAMTile | Design Tools |
|---|---|---|---|---|---|---|---|---|
| Kintex-Ultrascale+ | XCKU5PFFVB676-2E | 390.625 | 7665 | 12590 | 2200 | - | 3 | Vivado2019.1 |
| Zync-Ultarscale+ | XCZU28DR-FFVG1517-2-E | 390.625 | 7665 | 12588 | 2280 | - | 3 | Vivado2019.1 |

Notes:

1) Actual logic resource dependent on percentage of unrelated logic

## Applications

Typically, as the Bit Error Rate (BER) increases in high-speed networking systems such as 25G Ethernet, high-speed Ethernet specifications often require the integration of Forward Error Correction (FEC) into the standard. To address this requirement, Design Gateway has developed the XXVGMACRSFEC IP, which provides a lightweight and low-latency solution for FPGA-based systems implementing 25G Ethernet applications with RS-FEC feature.

The IP is designed to be fully compliant with Design Gateway's Network IP suite, which includes the TOE25G IP and UDP25 IP. This affords the system a complete solution for implementing TCP/IP and UDP/IP protocols based on 25G Ethernet with FEC. Consequently, implementing a full network stack of TCP/IP, UDP/IP, and Ethernet data streaming can be achieved without the need for CPU and external memory usage.

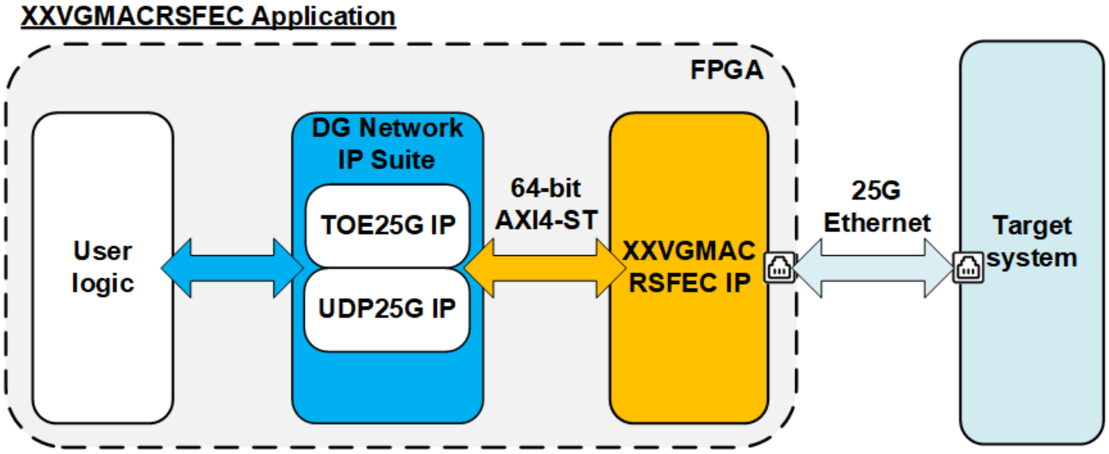


**Figure 1: XXVGMACRSFEC Application**

If the RS-FEC feature is unnecessary for the system, Design Gateway provides an alternative solution for Ethernet MAC IP core, called 10G25G EMAC IP. More details about this IP can be found on our website.
https://dgway.com/products/IP/GEMAC-IP/dg_10g25gemacip_data_sheet_xilinx.pdf

## Reference design

The XXVGMACRSFEC IP was launched with a loopback demo design on a Xilinx development board. The demo involves the Test logic generating and sending data packets to the IP, and then waiting for their return through SFP28 loopback module to the Test logic. The measured latency time was the time that the Test logic waited for the packets, which included the latency time within the IP, Xilinx transceiver, and SFP28 loopback module. As shown in Figure 2, the XXVGMACRSFEC IP shows an average round-trip latency of 473.60 ns, measured from the transmit path to the receive path of the XXVGMACRSFEC IP's user interface using the start-of-frame to start-of-frame at a frequency of 390.625 MHz. Specifically, the transmit latency time of the XXVGMACRSFEC IP is 61.44 ns, while the receive latency time is 360.93 ns. Consequently, the latency time of the GTY transceiver of UltraScale+ device is 51.23 ns.

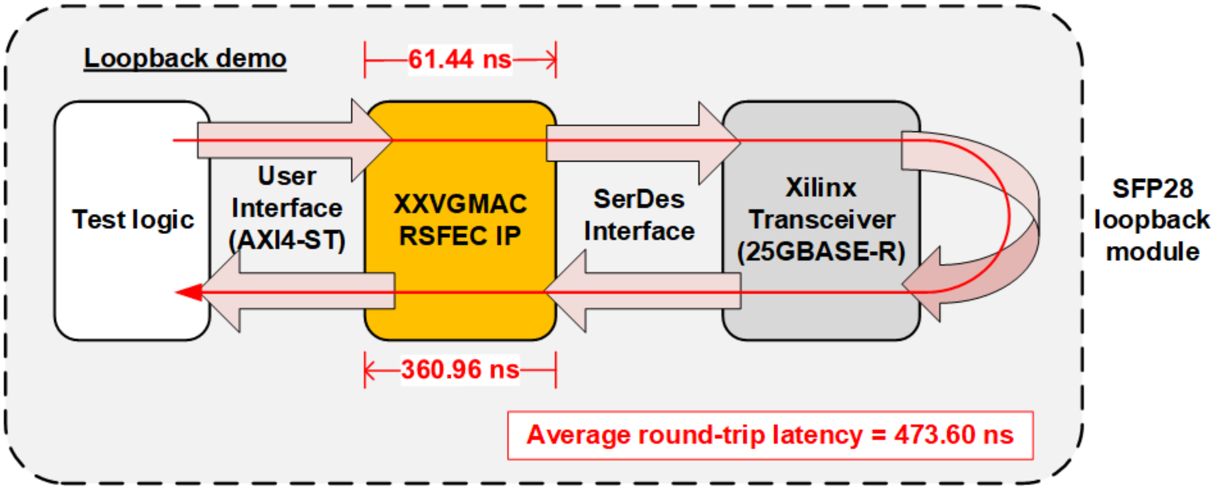


**Figure 2: XXVGMACRSFEC reference design**

## General Description

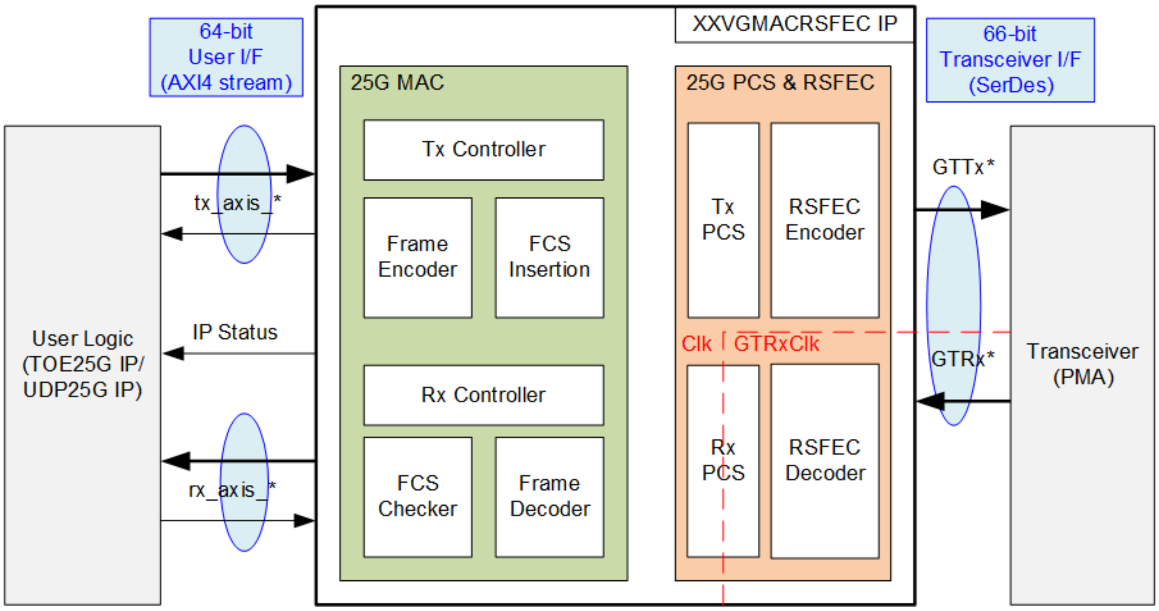


**Figure 3: XXVGMACRSFEC IP Block Diagram**

XXVGMACRSFEC IP provides a complete solution for 25G Ethernet applications with RS-FEC capabilities, implementing the 25G MAC, PCS, and RS-FEC. The design focuses on minimizing latency and resources while ensuring ease-of-use. The user interface of the IP is a 64-bit AXI4 stream operating at a single clock domain of 390.625 MHz, making it simple to use. On the other hand, the IP connects to the Xilinx transceiver through a 66-bit SerDes interface, which operates using two clock domains, Clk for GTTx signals and GTRxClk for GTRx signals, as shown in Figure 3.

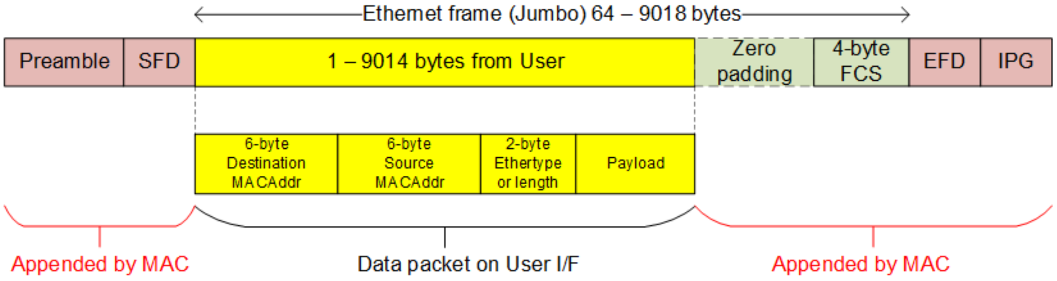


**Figure 4: Ethernet packet and IP's user packet structure**

In Figure 4, the 25G MAC function is illustrated, which is responsible for creating an 802.3 Ethernet frame from the user packet transmitted via the tx_axis_* signals. The 25G MAC computes a 4-byte Frame Check Sequence (FCS) and adds it as the footer to the Ethernet frame. Additionally, it supports the Jumbo frame feature, which allows for transferring packets up to 9018 bytes in size. If the transmitted user packet size is less than 60 bytes, the 25G MAC generates Zero-padding to extend the frame size since the minimum Ethernet frame size is 64 bytes. The user packet size can be configured within a range of 1-9014 bytes.

To form the transmitted packet, the Ethernet frame is supplemented with the preamble, Start frame delimiter (SFD), and End frame delimiter (EFD), followed by the FCS. Additionally, the Interpacket gap (IPG) is inserted to guarantee a minimum of a 40-bit gap between successive Ethernet packets.

The Tx PCS function is to convert the 64-bit Ethernet packet to 66-bit line code by 64B/66B encoding, followed by scrambling the data. The scrambled data is then fed to the RS-FEC encoder to calculate and append RS-FEC parities, forming the RSFEC encoded block. Finally, the encoded block is serialized to the 66-bit SerDes I/F of the transceiver.

On the receiving side, the 66-bit data received from the SerDes I/F is first de-serialized before being input to the RSFEC decoder. If an error is detected, the decoder corrects the decoded data and removes the added parities within the code block. The decoded data is then passed on the Rx PCS for descrambling and 64B/66B decoding, resulting in an Ethernet packet. This Ethernet packet is then transferred to the 25G MAC, where the header and footer (preamble, SFD, and EFD) are removed to obtain the Ethernet frame. The 25G MAC validates the FCS of the Ethernet frame and forwards the Ethernet frame to the user interface, excluding the 4-byte FCS. If an incorrect FCS is detected, the IP activates the error signal through the indicator of the user interface

The XXVGMACRSFEC IP employs a single clock domain in its user interface, enhancing user convenience. The RSFEC decoder and some logics in the Rx PCS operate on a different clock from the remaining logics. The Rx PCS includes logic that handles the asynchronous issue.

Notably, the XXVGMACRSFEC IP does not contain a transmit buffer, so the data input stream for sending user packet must be valid for every clock cycle during the transfer. Similarly, the IP does not have a receive buffer, and as a result, the received packet is continuously forwarded to the user via the AXI4 stream interface without interruption.

Additionally, XXVGMACRSFEC IP supports the zero-padding function, which facilitates sending user packets without considering the minimum packet size since the minimum value is 1 byte. However, the IP does not remove the zero-padding in the received user packet. Therefore, user needs to consider whether the received user packet has been padded with zero or not.

## Functional Description

Figure 3 shows that the XXVGMACRSFEC IP comprises of three primary layers: the MAC layer, the PCS layer, and the RS-FEC layer. Each layer will be discussed in details in the subsequent subtopics.

### MAC layer

To enable parallel operation of transmitted and received packets, the MAC layer employs separate logic blocks. For each transfer direction, three submodules have been integrated: FCS processer, Ethernet frame processer, and the controller.

- **FCS Insertion**

  Each user packet through Tx user interface (tx_axis_*) is input for CRC-32 calculation. The FCS is created using the polynomial of CRC-32 in accordance with the IEEE802.3 standard.

  $P(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X^{1} + 1$

  Once the calculation is complete, the FCS is added to the transmit user packet as the footer of the transmitted Ethernet frame after the user packet.

- **Frame Encoder**

  This module adds a zero-padding field to the transmit user packet if the packet size is less than 60 bytes. It then inserts the packet header and footer, consisting of a 7-byte preamble, 1-byte SFD, 4-byte FCS, and 1-byte EFD. Finally, the Idle code, which range in size from 11-18 bytes, is inserted as an IPG between two Ethernet packets.

- **Tx Controller**

  The controller monitors control signals on the Tx user interface. When a new transmit user packet is detected, it sends the control signal to the Frame Encoder module to insert the header and footer to create the Ethernet packet. The controller then de-asserts the ready signal on the Tx user interface for 3 or 7 clock cycles to complete FCS, EFD and IPG insertion process if the user packet size is 60 bytes or greater. The time for de-asserting the ready signal is determined by the IP for bandwidth compensation of the physical layer, typically de-asserting for 3 clock cycles. When the user packet size is less than 60 bytes, zero-padding is appended, which also causes the ready on the Tx user interface to be de-asserted. Therefore, the ready signal on the Tx user interface is de-asserted for more than 3 or 7 clock cycles in this case. Please refer to the Transmit interface timing diagram for further details.

  If a user packet with unsupported features is transmitted, an error will be generated. These features include de-asserting valid before the end of the packet, packets larger than 9014 bytes, and de-asserting byte tenable of some data before the end of the packet.

- **FCS Checker**

  This module utilizes the same CRC-32 calculation logic as the FCS Insertion module, but instead of inserting FCS into the Ethernet frame, it verifies the FCS extracted from the frame. If the FCS does not match, an error is asserted on Rx user interface (rx_axis_*).

- **Frame Decoder**

  This module decodes the data and control signals from the Rx PCS, monitors link up status, tracks the SFD and EFD of received Ethernet packet, removes the header and footer of the packet, re-aligns the data packet, and forwards the resulting user packet without the 4-byte FCS field on the Rx user interface. The first data byte on the Rx user interface is always located at rx_axis_data[7:0].

- **Rx Controller**

  The controller checks the SFD and FCS of the received Ethernet packet, asserts the data valid to the Rx user interface when the packet header is correct, and asserts an error when the FCS of the packet is incorrect.

### PCS layer

The PCS layer is responsible for implementing the Physical coding sublayer for 25GBASE-R PHY. It serves as an interface between the Physical Medium Attachment (PMA) sublayer and the Media-Independent Interface (MII), which is the MAC layer. The functionality of this layer is designed in accordance with the IEEE 802.3 ae specification and can be divided into a Tx PCS and Rx PCS as follows.

- **Tx PCS**

  This block performs a 64B/66B encoding of the 64-bit input data received from MAC layer. The resulting 66-bit line code data is then scrambled using the polynomial

  $$P(x) = x^{58} + x^{39} + 1$$

  before being transmitted to the RS-FEC layer.

- **Rx PCS**

  It receives 66-bit line code data from the RS-FEC layer. Initially, the data is used to detect block synchronization. After synchronization is complete, the Bit Error Rate (BER) function is enabled to detect incorrect data. The 66-bit line code data is then descrambled using the same polynomial as Tx PCS block. Finally, the data is decoded by 64B/66B decoding function to produce 64-bit data, which is forwarded to the MAC layer.

### RS-FEC layer

The RS-FEC layer provides support for the Reed-Solomon Forward Error Correction sublayer for 25GBASE-R PHY, as specified in IEEE 802.3 clause 108 specification. This layer implements a Reed-Solomon code with RS(528, 514) operating over the Galois Field GF($2^{10}$), where the symbol size is 10 bits, and the polynomial is defined as

$$P(x) = x^{10} + x^3 + 1$$

The RS-FEC layer has different functionalities for the transmitter (RSFEC Encoder) and receiver (RSFEC Decoder), which are as follows.

- **RSFEC Encoder**
  Upon receiving data from Tx PCS, a process is initiated to remove a certain amount of IPG from the data in order to insert a codeword markers into the stream. The resulting data and codeword marker are then combined to form a RS code block, which is used to calculate and append RS-FEC parities. Once this is done, the code block, along with the RS-FEC parities, is forwarded to the Transceiver.
- **RSFEC Decoder**
  Once the codeword marker has been detected and marked, the operation starts. First, the RS code block is decoded and any corrupt data is corrected if needed. Next, the code block is separated into the data, codeword marker, and RS-FEC parities. The RS-FEC parities are removed, and the codeword marker is replaced by the IPG to ensure the rate is maintained. Finally, the data and the IPG are merged to form 66-bit line code data that is sent to Rx PCS.
  *Note: The RS-FEC decoder is designed for Error indicator bypass mode. Please contact our sales if fully decoding or other bypass modes are required.*

## User Logic

For transferring data by using TCP/IP protocol or UDP/IP protocol, the user interface of the XXVGMACRSFEC IP can be directly connected with DG TOE25G IP for TCP/IP solution or DG UDP25G IP for UDP/IP solution on 25Gb Ethernet.

More details of the IP are described in the datasheet.

https://dgway.com/products/IP/UDP25G-IP/dg_udp25gip_data_sheet_xilinx.pdf

https://dgway.com/products/IP/TOE25G-IP/dg_toe25gip_data_sheet_xilinx_en.pdf

## 25G Ethernet PMA (25G BASE-R)

Xilinx’s GTY transceiver has been designed to support the 25G Ethernet PMA for 25G BASE-R. To configure the transceiver parameters for the operation of the 25G PMA, Xilinx provides the UltraScale FPGAs Transceivers Wizard. An example configuration of the Xilinx GTY transceiver parameter on the KCU116 board is provided below.

| | |
|---|---|
| • Line Rate (Gb/s) | : 25.78125 |
| • PLL type | : QPLL0 |
| • Requested reference clock (MHz) | : 156.25 |
| • Actual Reference Clock (MHz) | : 322.265625 |
| • Encoding/Decoding | : Async. gearbox for 64B/66B |
| • User data width | : 64 |
| • Internal data width | : 64 |

For further information on the Transceivers Wizard and GTY Transceiver, please refer to the link provided.
https://www.xilinx.com/products/intellectual-property/ultrascale_transceivers_wizard.html
https://www.xilinx.com/content/dam/xilinx/support/documents/user_guides/ug578-ultrascale-gty-transceivers.pdf

## Core I/O Signals

Descriptions of all I/O signals are provided in Table 2 - Table 3.

### Table 2: User I/O Signals (Synchronous to Clk signal)

| Signal | Dir | Description |
|---|---|---|
| **Common Interface** | | |
| RstB | In | Reset IP core. Active Low. |
| Clk | In | Fixed clock frequency input, synchronous with Tx PMA interface (390.625 MHz for 25G Ethernet). |
| **User interface (AXI4 Stream I/F)** | | |
| tx_axis_tdata[63:0] | In | Transmitted data. Valid when tx_axis_tvalid is asserted to 1b. |
| tx_axis_tkeep[7:0] | In | Byte enable of 64-bit tx_axis_tdata. Each bit is set to 1b for each valid byte. Bit[0],[1],[2],…,[7] represent the valid byte of tdata[7:0],[15:8],[23:16],…,[63:56], respectively. To enable all data in the packet, except the last word, this signal must be set to 0xFF. However, for the last word, the data may only be valid for some bytes, and byte enable for the last data can set by one of eight values: 0x01, 0x03, 0x07, 0x0F, 0x1F, 0x3F, 0x7F, and 0xFF, depending on how many bytes are valid. The signal is only valid when tx_axis_tvalid is set to 1b. |
| tx_axis_tvalid | In | Transmitted data valid signal. Asserts to 1b to indicate that tx_axis_tdata/tkeep/tlast/tuser are valid. |
| tx_axis_tlast | In | Asserts to 1b to indicate the final word in the frame.<br>Valid when tx_axis_tvalid is asserted to 1b. |
| tx_axis_tuser | In | Asserts to 1b to cancel the current transmitted frame.<br>Valid at the end of packet (tx_axis_tlast=1b and tx_axis_tvalid=1b). |
| tx_axis_tready | Out | Handshaking signal. Asserted to 1b when the data on tx_axis_tdata has been accepted. It remains asserted to 1b while transmitting a packet to ensure continuous transfer of the packet. |
| rx_axis_tdata[63:0] | Out | Received data. Valid when rx_axis_tvalid is asserted to 1b. |
| rx_axis_tkeep[7:0] | Out | Received data byte enable of 64-bit rx_axis_tdata. Each bit is set to 1b for each valid byte. Bit[0],[1],[2],…,[7] represent the valid byte of tdata[7:0],[15:8],[23:16],…,[63:56], respectively. The signal is valid when rx_axis_tvalid is asserted to 1b. |
| rx_axis_tvalid | Out | Received data valid signal. Asserted to 1b to indicate that rx_axis_tdata/tkeep/tlast/tuser are valid. |
| rx_axis_tlast | Out | Asserted to 1b to indicate the final word in the received frame.<br>Valid when rx_axis_tvalid is asserted to 1b. |
| rx_axis_tuser | Out | Asserted at the end of the received frame to indicate that the frame has an error.<br>0b: normal packet, 1b: error packet.<br>Valid when rx_axis_tlast and rx_axis_tvalid are asserted to 1b. |

| Signal | Dir | Description |
|---|---|---|
| **IP Status interface** | | |
| TestPin[47:0] | Out | Reserved to be IP Test point. |
| IPVersion[31:0] | Out | IP version number. |
| RxRSFECUser[7:0] | Out | RS-FEC receive status for user information<br>[0] – Asserted to 1b periodically when a RS-FEC block is received and processed.<br>[1] – Asserted to 1b when the received RS-FEC block has some errors, but those errors have been successfully corrected.<br>[2] – Asserted to 1b when the received RS-FEC block has some errors that cannot be corrected.<br>[7:3] – Reserved |
| TxUserErr[7:0] | Out | The error status of the transmit user interface, which becomes valid two clock cycles after the user sends the last data of the packet.<br>[0] – Asserted to 1b when tx_axis_tvalid is de-asserted to 0b while a packet is still transmitting.<br>[1] – Asserted to 1b when transmitting data that is not the last (tx_axis_tvalid=1b and tx_axis_tlast=0b) and tx_axis_tkeep[7:0] does not equal 0xFF.<br>[2] – Asserted to 1b when transmitting the last data and tx_axis_tkeep[7:0] does not equal one of the eight values: 0x01, 0x03, 0x07, 0x0F, 0x1F, 0x3F, 0x7F, and 0xFF.<br>[3] – Asserted to 1b when the user transmit packet size is greater than 9014 bytes.<br>[7:4] - Reserved |
| linkup | Out | Asserted to 1b once MAC layer initialization is finished. |

**Table 3: Physical Medium Attachment (PMA) I/O Signals**

| Signal | Dir | Description |
|---|---|---|
| **Tx PMA interface (Synchronous to Clk signal)** | | |
| GTTxHeader[1:0] | Out | Transmit header to the transceiver. |
| GTTxData[64:0] | Out | Transmit data to the transceiver. |
| **Rx PMA interface (Synchronous to GTRxClk signal)** | | |
| GTRxRstB | In | Reset IP core from Receiver of PMA interface. Active Low. |
| GTRxClk | In | Fixed clock frequency input, synchronous with Rx PMA interface<br>(390.625 MHz for 25G Ethernet).<br>*Note: GTRxClk and Clk signal are generated from separate clock sources, which cause slight differences in both frequency and phase between the two signals.* |
| GTRxHeader[1:0] | In | Receive header from the transceiver. |
| GTRxData[63:0] | In | Receive data from the transceiver. |
| GTRxGearboxSlip | Out | Asserted to slip the gearbox contents to the next possible alignment. |

## Timing Diagram

### IP Initialization

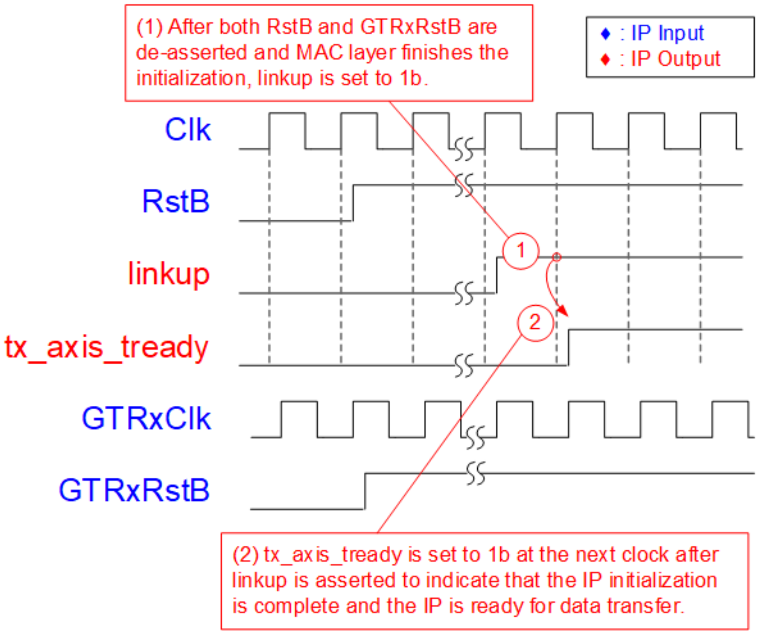

(1) After both RstB and GTRxRstB are de-asserted and MAC layer finishes the initialization, linkup is set to 1b.
♦ : IP Input
♦ : IP Output
Clk
RstB
linkup
tx_axis_tready
GTRxClk
GTRxRstB
1
2
(2) tx_axis_tready is set to 1b at the next clock after linkup is asserted to indicate that the IP initialization is complete and the IP is ready for data transfer.

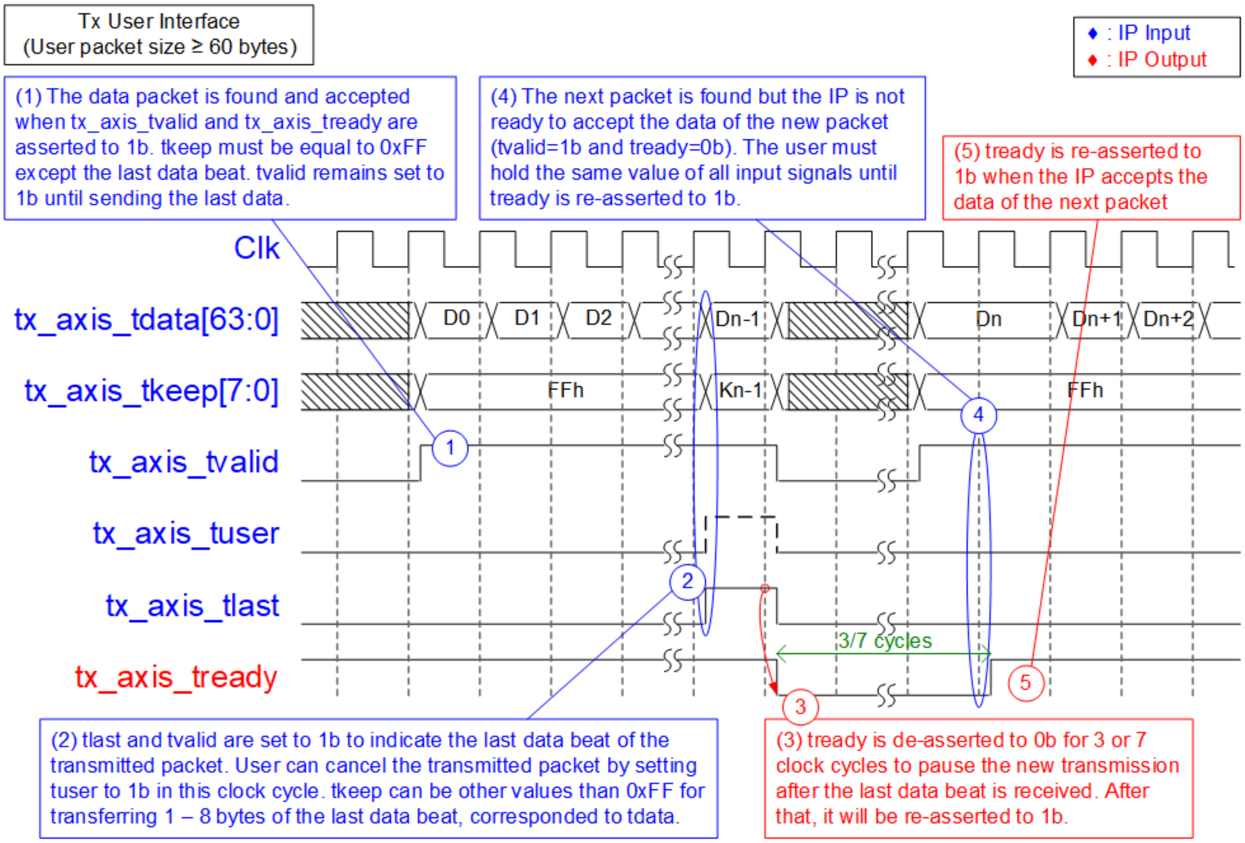
**Figure 5: Linkup status**

The sequence to assert linkup and tx_axis_tready when the system is initializing is as follows.

(1) The IP waits until both RstB and GTRxRstB are de-asserted. After that, it waits for a physical connection to finish initializing the MAC layer. Once the IP is initialized successfully, it sets the linkup signal to 1b to indicate that the Ethernet MAC is connected.

(2) After setting the linkup signal to 1b, the IP then sets the tx_axis_tready to 1b in the next clock cycle, indicating that it is ready for data transfer via the AXI4 stream (User) interface.

### Transmit User Interface



(1) The data packet is found and accepted when tx_axis_tvalid and tx_axis_tready are asserted to 1b. tkeep must be equal to 0xFF except the last data beat. tvalid remains set to 1b until sending the last data.

(4) The next packet is found but the IP is not ready to accept the data of the new packet (tvalid=1b and tready=0b). The user must hold the same value of all input signals until tready is re-asserted to 1b.

(5) tready is re-asserted to 1b when the IP accepts the data of the next packet

(2) tlast and tvalid are set to 1b to indicate the last data beat of the transmitted packet. User can cancel the transmitted packet by setting tuser to 1b in this clock cycle. tkeep can be other values than 0xFF for transferring 1 – 8 bytes of the last data beat, corresponded to tdata.

(3) tready is de-asserted to 0b for 3 or 7 clock cycles to pause the new transmission after the last data beat is received. After that, it will be re-asserted to 1b.

**Figure 6: Timing diagram of Tx User Interface to send user packet that is at least 60 bytes in size**

Figure 6 demonstrates the process of sending a user data packet via the Transmit user interface (64-bit AXI4 Stream). Assuming the user packet length is 60 bytes or greater, zero-padding is not required. Here are the steps in the data transfer process.

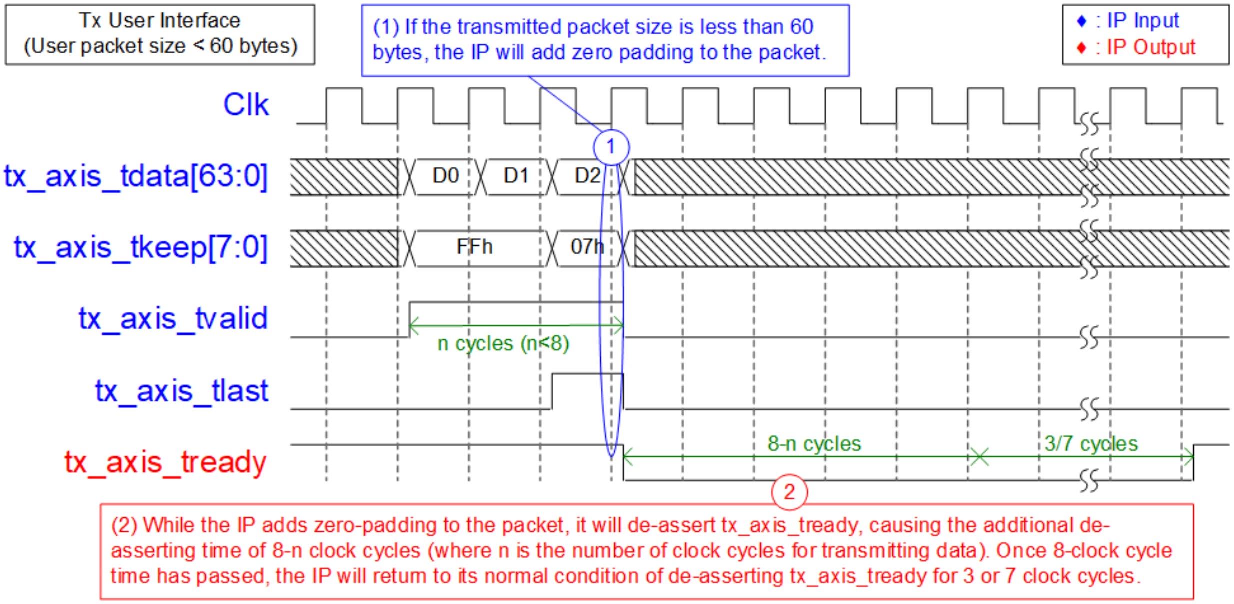
(1) The first data beat of the packet is transferred to the IP by setting tx_axis_tvalid to 1b. At the same time, the tx_axis_tready must also be set to 1b to indicate that the IP accepts this data. Once the first data is accepted, tx_axis_tvalid must remain set to 1b for continuous data transfer until the last data of packet is sent. The tx_axis_tkeep signal represents the number of valid bytes of the 8-byte data (tx_axis_tdata) and must be set to 0xFF to send 64-bit valid data for every transfer cycle, except the last data beat which allows sending the data that is valid only some bytes. The tx_axis_tready is also set to 1b to accept the continuous data until the packet transfer is complete.

(2) The last data beat of a packet can be transferred by setting both tx_axis_tvalid and tx_axis_tlast to 1b. The tx_axis_tuser signal is used to allow the user to cancel the current data packet by setting it to 1b at the last data beat. In this cycle, tx_axis_tkeep can be set to a value other than 0xFF, allowing for the transfer of 1 – 8 bytes of tx_axis_tdata.

(3) One clock cycle after the IP receives the last data beat of the packet, it de-asserts tx_axis_tready to 0b to pause the data transmission. The tx_axis_tready is de-asserted to 0b for 3 or 7 clock cycles to create the Ethernet frame for this user data packet. After that, it is re-asserted to 1b, indicating that the IP is ready for the next data transmission. Typically, tx_axis_tready is de-asserted by the IP for 3 clock cycles. However, it may be de-asserted for 7 clock cycles if the IP needs to handle a specific task.

(4) If the user sends a new packet by setting tx_axis_tvalid to 1b, but the IP is still not ready to accept new data by de-asserting tx_axis_tready to 0b, all input signals of the AXI4 stream from user remain unchanged.

(5) Once the IP re-asserts tx_axis_tready to 1b, the first data is transferred, similar to step (1).



(1) If the transmitted packet size is less than 60 bytes, the IP will add zero padding to the packet.

(2) While the IP adds zero-padding to the packet, it will de-assert tx_axis_tready, causing the additional de-asserting time of 8-n clock cycles (where n is the number of clock cycles for transmitting data). Once 8-clock cycle time has passed, the IP will return to its normal condition of de-asserting tx_axis_tready for 3 or 7 clock cycles.

**Figure 7: Timing diagram of Tx User Interface to send user packet with a size below 60 bytes**

Figure 7 demonstrates the user of the IP's zero-padding functionality for a small data packet with a size below 60 bytes.

(1) If the packet size is below the minimum frame size (60 bytes), the IP will receive the last beat of the packet and add "zero-padding" data to it. In Figure 7, the IP receives 3 data beats, 19 bytes (8+8+3 bytes), requiring the IP to append 41 bytes (60 – 19 bytes) of zero-padding to create a complete Ethernet frame.

(2) While performing the zero-padding feature, the IP is unable to receive new data. The IP requires 8 clock cycles to create the minimum Ethernet frame. As a result, the IP will de-assert tx_axis_tready for an additional 8 – n clock cycles, where n represents the number of clock cycles used to transfer the user packet. After this, the normal de-asserting time of 3 or 7 clock cycles is resumed.

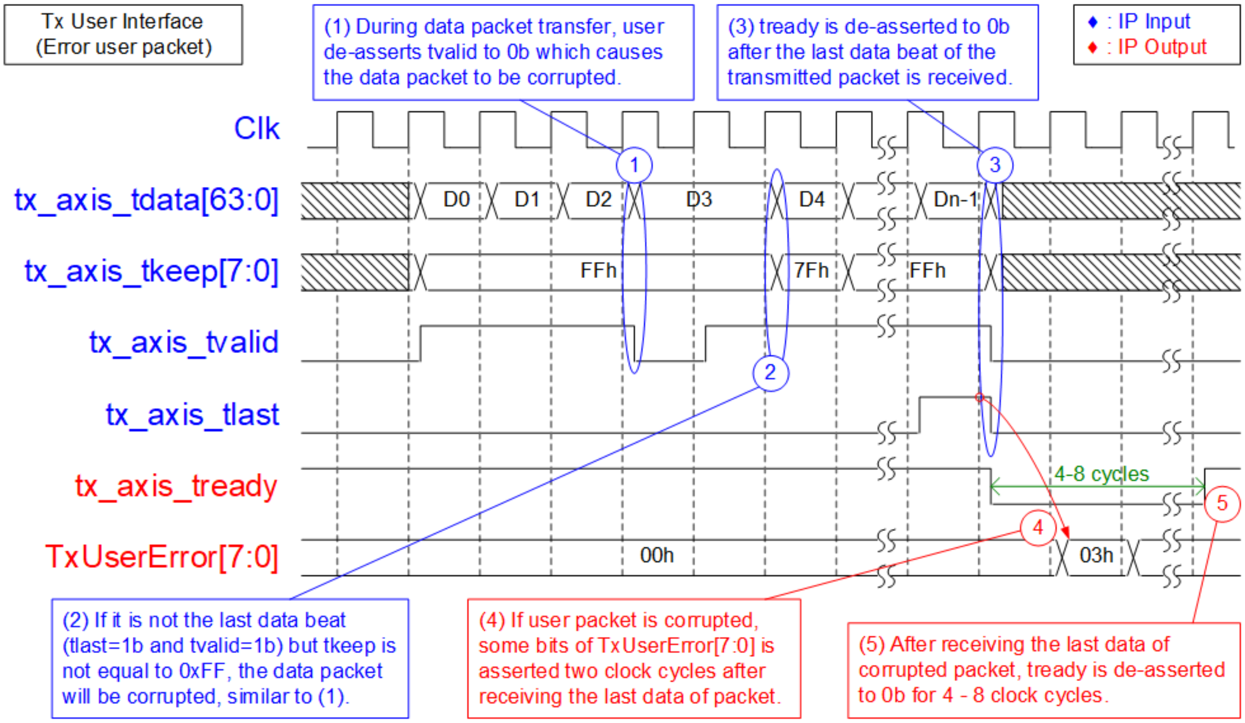

Tx User Interface
(Error user packet)
(1) During data packet transfer, user de-asserts tvalid to 0b which causes the data packet to be corrupted.
(3) tready is de-asserted to 0b after the last data beat of the transmitted packet is received.
♦ : IP Input
♦ : IP Output
Clk
tx_axis_tdata[63:0]
tx_axis_tkeep[7:0]
tx_axis_tvalid
tx_axis_tlast
tx_axis_tready
TxUserError[7:0]
D0
D1
D2
D3
D4
Dn-1
FFh
7Fh
FFh
00h
03h
4-8 cycles
1
2
3
4
5
(2) If it is not the last data beat (tlast=1b and tvalid=1b) but tkeep is not equal to 0xFF, the data packet will be corrupted, similar to (1).
(4) If user packet is corrupted, some bits of TxUserError[7:0] is asserted two clock cycles after receiving the last data of packet.
(5) After receiving the last data of corrupted packet, tready is de-asserted to 0b for 4 - 8 clock cycles.

**Figure 8: Timing diagram of Tx User Interface when the user sends corrupted packet**

Figure 8 illustrates two error conditions resulting from the transmission of unsupported packets by the user. The first condition occurs when tx_axis_tvalid is de-asserted to 0b before the last data of the packet is transferred. The second condition arises when tx_axis_tkeep is set to FFh during the transfer of data that is not the last data of the packet. The actions by the IP upon detection of a transmit error are outlined below.

(1) When tx_axis_tvalid is de-asserted to 0b during a packet transfer, which violates the IP agreement, the IP discards the data packet by intentionally corrupting the FCS of the Ethernet frame.

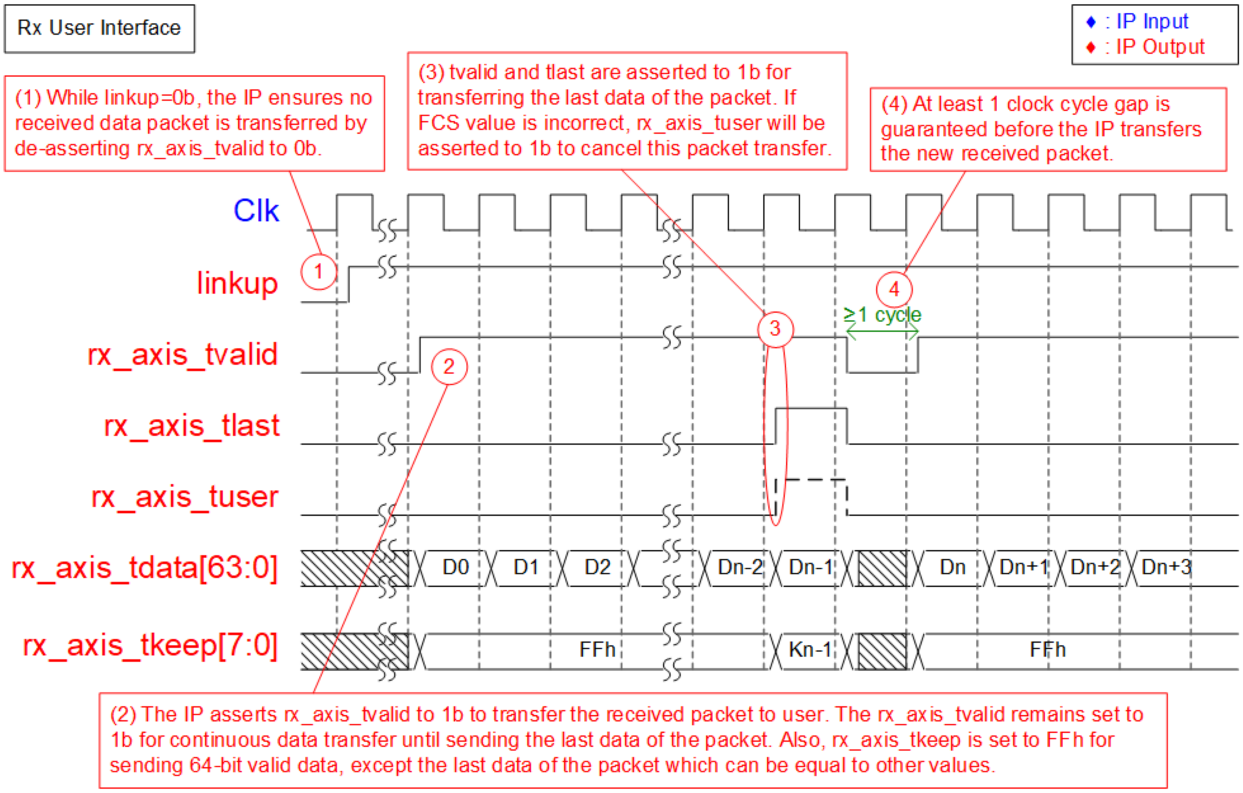
(2) Similar to (1), when tx_axis_tkeep is set to a value other than FFh for a non-last data beat leads to IP disagreement, the IP discards the data packet by intentionally corrupting the FCS of the Ethernet frame.

(3) The IP de-asserts tx_axis_tready to 0b after receiving the last data of the packet to initiate the error recovery process.

(4) Two clock cycles after receiving the last data of the packet, the IP asserts the error bits of TxUserError[7:0] to indicate the type of error in the packet, as described in Table 2.

(5) The tx_axis_tready is de-asserted to 0b to pause the data transmission for 4 to 8 clock cycles. Subsequently, it is re-asserted to 1b, indicating that the IP is ready for the next data transmission.

### Receive Interface



(1) While linkup=0b, the IP ensures no received data packet is transferred by de-asserting rx_axis_tvalid to 0b.

(3) tvalid and tlast are asserted to 1b for transferring the last data of the packet. If FCS value is incorrect, rx_axis_tuser will be asserted to 1b to cancel this packet transfer.

(4) At least 1 clock cycle gap is guaranteed before the IP transfers the new received packet.

(2) The IP asserts rx_axis_tvalid to 1b to transfer the received packet to user. The rx_axis_tvalid remains set to 1b for continuous data transfer until sending the last data of the packet. Also, rx_axis_tkeep is set to FFh for sending 64-bit valid data, except the last data of the packet which can be equal to other values.

**Figure 9: Timing diagram of Receive User Interface packet timing diagram**

The timing diagram of the Receive user interface, AXI4 stream, upon receiving a new packet is shown in Figure 9, and additional information is provided below.

(1) If the Ethernet connection is not successfully established (linkup=0b), the rx_axis_tvalid is always set to 0b, indicating that no data packet has been received.

(2) The IP asserts rx_axis_tvalid to 1b to transfer the first data of the received packet. The rx_axis_tvalid remains set to 1b to transfer the packet continuously until the last data of the packet is sent. The rx_axis_tkeep is always set to FFh, enabling the transfer of all 8 bytes of rx_axis_tdata, except for the last data, which may contain fewer valid bytes.
*Note: The IP does not remove zero-padding from the received packet.*

(3) If the FCS of the received Ethernet frame is incorrect, the rx_axis_tuser is set to 1b during the last data transfer, as indicated by the rx_axis_tvalid=1b and rx_axis_tlast=1b. Normally, the rx_axis_tuser should be set to 0b.

(4) After the IP sends the last data, the rx_axis_tvalid is de-asserted to 0b for at least one clock cycle before a new received packet is transferred.

### IP Status Interface for RS-FEC

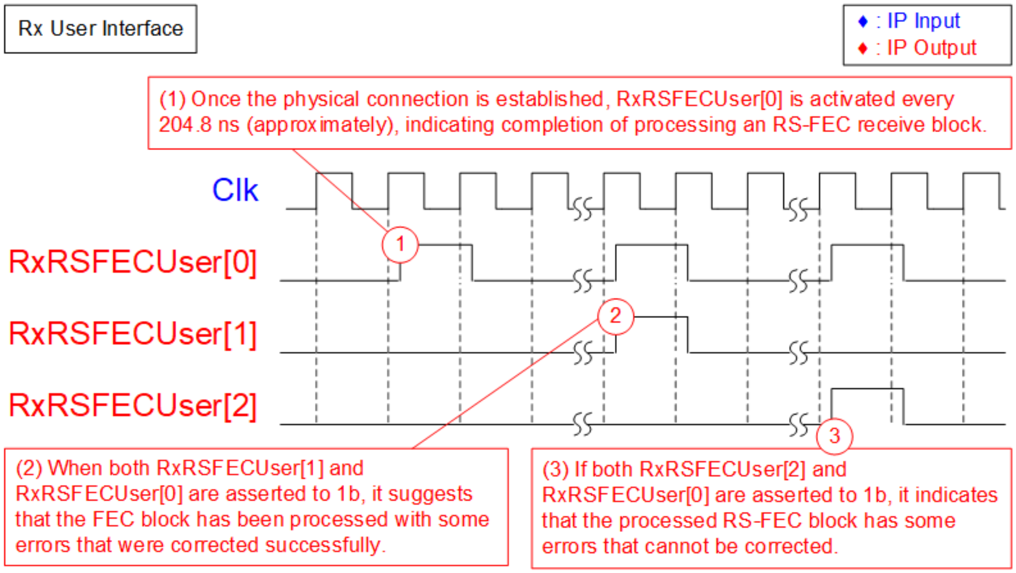

Rx User Interface
♦ : IP Input
♦ : IP Output
(1) Once the physical connection is established, RxRSFECUser[0] is activated every 204.8 ns (approximately), indicating completion of processing an RS-FEC receive block.
Clk
RxRSFECUser[0]
RxRSFECUser[1]
RxRSFECUser[2]
1
2
3
(2) When both RxRSFECUser[1] and RxRSFECUser[0] are asserted to 1b, it suggests that the FEC block has been processed with some errors that were corrected successfully.
(3) If both RxRSFECUser[2] and RxRSFECUser[0] are asserted to 1b, it indicates that the processed RS-FEC block has some errors that cannot be corrected.

**Figure 10: Timing diagram of IP Status interface for RS-FEC operation**

Figure 10 shows the status interface for the RS-FEC receiver operation, with further information provided below.

(1) Once the physical connection is established, the RxRSFECUser[0] is periodically asserted to 1b every 204.8 ns (approximately), indicating that the RS-FEC receiver is processed.

(2) If the RS-FEC receiver detects some errors that are successfully corrected, both RxRSFECUser[0] and RxRSFECUser[1] are asserted to 1b. This status can be utilized to check the physical connection quality. The data output from the IP is still correct.

(3) In case the RS-FEC receiver detects uncorrectable errors, both RxRSFECUser[0] and RxRSFECUser[2] are asserted to 1b. This could result in the IP not functioning properly, potentially leading to a connection reset as Bit Error Rate (BER) in the system increases.

## Verification Methods

The 25GEMAC/PCS+RS-FEC IP Core functionality was verified by simulation and also proved on real board design by using KCU116 and ZCU111 evaluation board.

## Recommended Design Experience

User must be familiar with HDL design methodology to integrate this IP into the design.

## Ordering Information

This product is available directly from Design Gateway Co., Ltd. Please contact Design Gateway Co., Ltd. for pricing and additional information about this product using the contact information on the front page of this datasheet.

## Revision History

| Revision | Date | Description |
|---|---|---|
| 1.0 | Mar-31-2023 | New release |