OpenCores.Org

10GE MAC Core Specification

Author: A. Tanguay

antanguay@opencores.org

Rev. 0.8

11/25/12

This page has been intentionally left blank.

Revision History

Rev.	Date	Author	Description
0.1	5/30/2008	A.Tanguay	Draft
0.2	6/6/2008	A.Tanguay	Added block diagram. Added mod[2:0] signals.
0.6	05/23/2009	A. Tanguay	Added details of operation.
0.7	12/13/2009	A. Tanguay	Added big endian mode for packet interface.
0.8	11/25/2012	A. Tanguay	Added basic packet statistics. Added application example.

Contents

1. Introduction

The 10GE MAC core is designed for easy integration with proprietary custom logic. It features a POS-L3 like interface for the datapath and a Wishbone compliant interface for management. The core was intentionally designed with a limited feature set for a small gate footprint.

2. Architecture

A block diagram of the core is presented in the figure below.

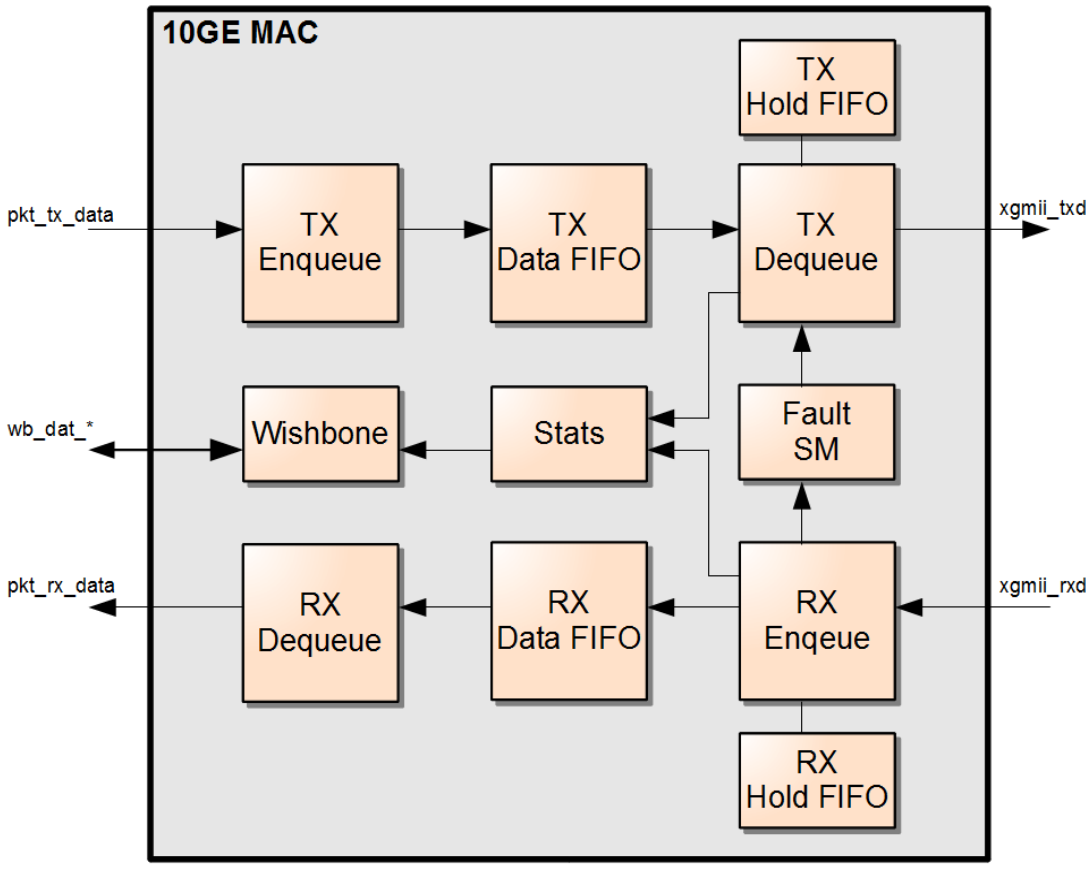

Figure 1: Block Diagram

A design example is shown below where the MAC interfaces to a XAUI macro built into the FPGA. In this example, an external PHY is used to convert from XAUI to a 10.3125Gbps signal suitable XFP and SFP+ modules.

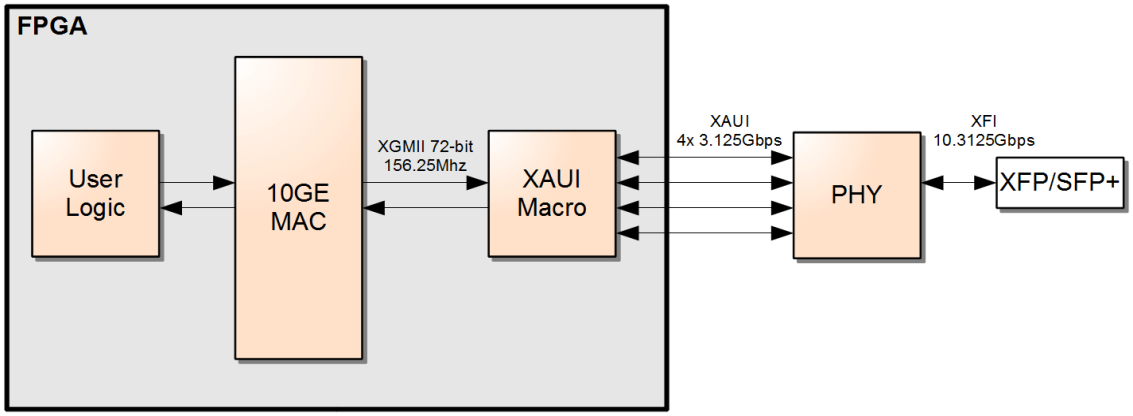

Figure 2: Design Example

3. Operation

TX Enqueue Engine

The TX Enqueue Engine receives frames from the user's core logic and stores them in the transmit FIFO along with some additional flags such as SOP and EOP indicators. It also provides FIFO fill status (available signal) to the core.

TX FIFO

The TX FIFO is 128-entry deep by default with 64-bit of data and 8-bit of status per entry. Since the FIFO can only store 1024 bytes of data (128 x 64-bit), the MAC must operate in flow-through mode, meaning that the transmission of a frame on the XGMII interface can (and must) start while the frame is still being written to the FIFO on the enqueue side.

The upper bits of the FIFO, bits 71:64, contains status information used by the TX Dequeue Engine to maintain frame alignment. The format of the FIFO is shown in the next figure:

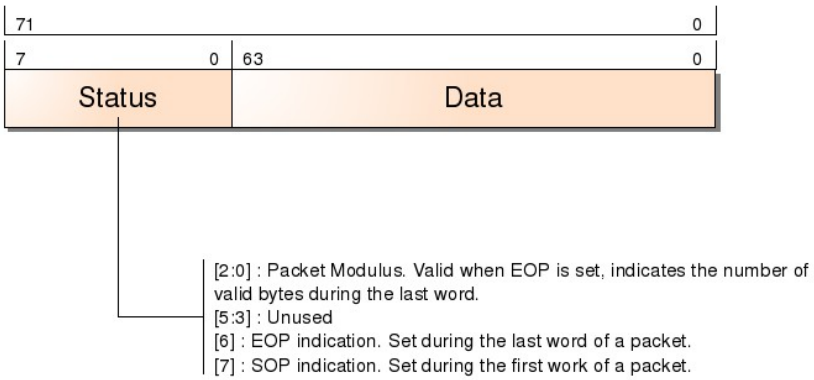

Figure 3: Transmit FIFO Format

TX Dequeue Engine

The TX Dequeue Engine contains two state-machines. The first state-machine reads data from the data FIFO and pads small packets to the minimum 64-byte required for Ethernet. The state-machine writes data to the holding FIFO and the CRC calculation logic. The purpose of the holding FIFO is to compensate for latency in the CRC calculation. The CRC logic operates in 64-bit data at the exception of the last word of the frame. Since the last word may contain 1 to 8 valid bytes, the CRC logic transitions to 8-bit mode at the end of the frame. Up to 8 cycles may be required to complete the calculation, hence the 8-word delay in the Holding FIFO.

The Encoding State-Machine reads data from the Holding FIFO after inserting the Ethernet preamble. To minimize the logic, this state-machine aligns all data on 64-bit boundary and generates flags indicating when 32-bit alignment is necessary to meet the required IFG. The IFG is calculated based on the accumulated Deficit Idle Count (DIC) and modulus of the current frame. Resulting flags are passed to the Barrel Shifter which performs the final 32-bit alignment.

The RC Layer monitors link status signals from the Fault State-Machine and inserts remote fault messages when a local fault was detected. The fault signal is also passed to the Encoding State-Machine which stops transmitting packets during faults.

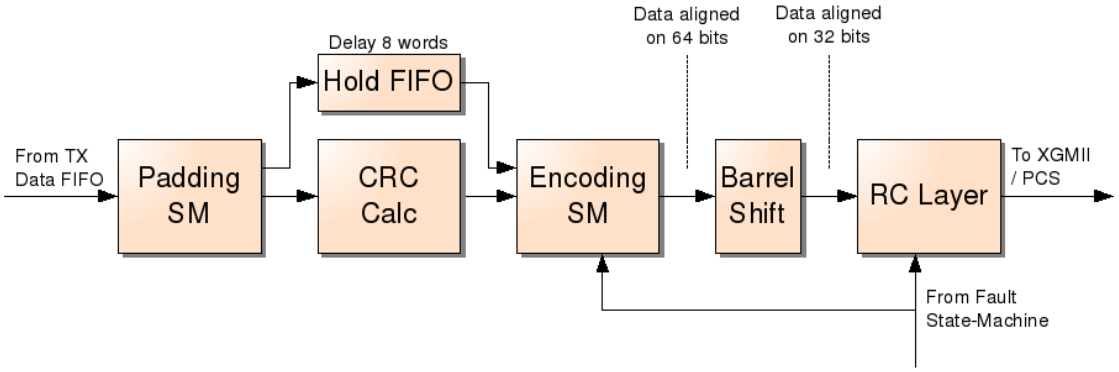

Figure 4: TX Dequeue Engine

RX Enqueue Engine

In the RX Enqueue Engine, the RC layer monitors the XGMII interface for fault conditions and pass the status to the Fault State-Machine. The Barrel Shifter looks for the start of frame delimiters on 32-bit boundary and re-aligns the data on 64-bit boundary. This method reduces the complexity of the next stages.

The Decoding State-Machine has the complex task of delimiting frames and detecting invalid frames such as fragments and runts. As it decodes the data, it write a copy to the

Holding FIFO and CRC logic. The purpose of the holding FIFO is to compensate for latency in the CRC calculation. The CRC logic operates in 64-bit data at the exception of the last word of the frame. Since the last word may contain 1 to 8 valid bytes, the CRC logic transitions to 8-bit mode at the end of the frame. Up to 8 cycles may be required to complete the calculation, hence the 8-word delay in the Holding FIFO.

As data emerges from the Holding FIFO, it is written to the RX Data FIFO along with SOP, EOP and other flags. If the CRC logic reports an error during that time, an error flag is set in the next FIFO entry.

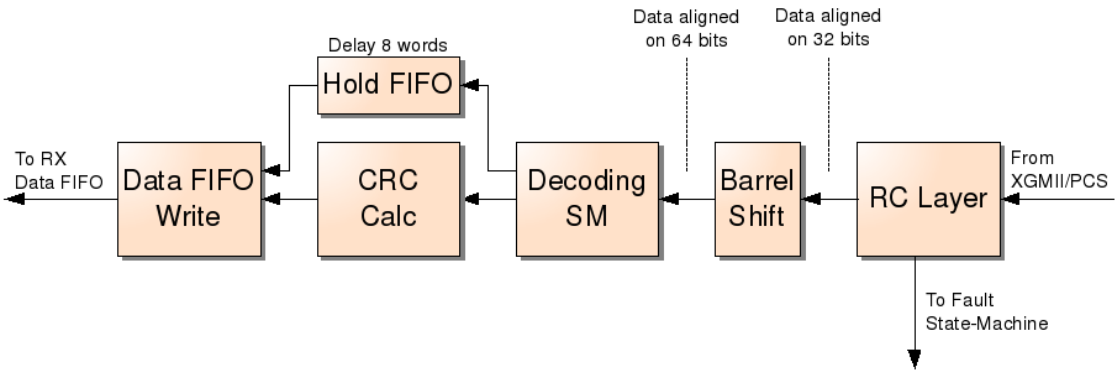

Figure 5: RX Enqueue Engine

RX FIFO

The RX FIFO is 128-entry deep by default with 64-bit of data and 8-bit of status per entry. Since the FIFO can only store 1024 bytes of data (128 x 64-bit), the MAC must operate in flow-through mode, meaning that the transmission of a frame on the packet interface can (and must) start while the frame is still being written to the FIFO on the enqueue side.

The upper bits of the FIFO, bits 71:64, contains status information used by the TX Dequeue Engine to maintain frame alignment. Also includes is an error flag used to propagate CRC errors or any other error conditions to the Dequeue Engine. The format of the FIFO is shown in the next figure:

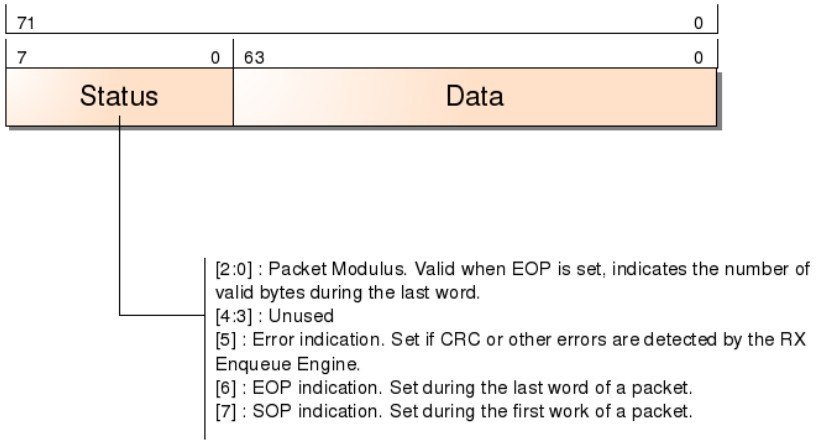

Figure 6: RX FIFO Format

RX Dequeue Engine

The RX Dequeue Engine interfaces with the user logic. Its primary function is to convert the internal status from the RX FIFO into meaninful signals on the pkt_rx interface.

Fault State-Machine

The Fault State-Machine monitors fault messages captured by the RX Enqeue Engine. It's role is to debounced fault indications and transition the fault state according to the 802.3ae speciifcation. The state-machine can declare two types of faults: a local fault or a remote fault. Both fault types can be monitored by reading the interrupt status register.

A local fault is usually the result of the PHY's inability to detect valid 64B/66B sequences on the incoming 10.3125Gbps signal.

A remote fault is asserted by the far-end PHY and typically means that it failed to detect valid 64B/66B sequences.

Stats Engine

The Stats Engines monitors frame indication signals from the TX Dequeue and RX Enqeue Engines and accumulates various packet and byte counters. It also is used to traverse from the high-speed clock domains to the lower-speed processor interface clock domain.

Registers

Register List

Name	Address	Width	Access	Description
Configuration Register 0	0x00	32	R/W	
Interrupt Pending Register	0x08	32	COR	
Interrupt Status Register	0x0c	32	RO	
Interrupt Mask Register	0x10	32	R/W	
Transmit Octets Count	0x80	32	RO	
Transmit Packet Count	0x84	32	RO	
Receive Octets Counts	0x90	32	RO	
Receive Packet Count	0x94	32	RO	

Configuration Register 0 – 0x00

The registers of the WISHBONE slave unit start at offset 0x180.

RESET VALUE: 0x00000001

Bit #	Name	Description
31 - 1	Reserved	Reserved field. Reading returns zeros.
0	TX Enable	When set, transmission of frames is enabled. This bit is set by default.

Table 1: Configuration Register 0

Interrupt Pending Register – 0x08

This register is clear-on-read and provides an indication of pending interrupts.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 9	Reserved	Reserved field. Reading returns zeros.
8	RX Fragment Error	When set, indicates that a fragment was received. A fragment is a frame that has a start of frame delimiter but no end of frame. Typically it is immediately followed by a new start of frame.
7	RX CRC Error	When set, indicates that a frame was received with invalid CRC.
6	RX Pause Frame	When set, indicates that a pause frame was received. Pause frames are discarded.
5	Remote Fault	When set, indicates that the fault state-machine has transitioned in or out of remote fault condition.
4	Local Fault	When set, indicates that the fault state-machine has transitioned in or out of local fault condition.
3	RX Data Fifo Underflow	When set, indicates that the receive data FIFO has experienced underflow. This may occur if "pkt_rx_ren" is asserted when no data is available in the receive FIFO.
2	RX Data Fifo Overflow	When set, indicates that the receive data FIFO has experienced overflow. This may occur if "pkt_rx_ren" remains de-asserted for a too long period while data is present in the receive FIFO.
1	TX Data Fifo Underflow	When set, indicates that the transmit data FIFO has experienced underflow. This situation should never occur.
0	TX Data Fifo Overflow	When set, indicates that the transmit data FIFO has experienced overflow. This may occur if "pkt_tx_val"is asserted and no room is left in the transmit FIFO.

Table 2: Interrupt Pending Register

Interrupt Status Register – 0x0C

This register is read-only and provides a direct status of interrupt lines used to set bits in the Interrupt Pending Register.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Interrupt Status	Each bit corresponds to an interrupt in the Interrupt Pending Register.

Table 3: Interrupt Status Register

Interrupt Mask Register – 0x10

This register is read/write and can be used to enable individual interrupts. By default all interrupts are disabled.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Interrupt Mask	Each bit corresponds to an interrupt in the Interrupt Pending Register. Setting a bit enables the interrupt.

Table 4: Interrupt Mask Register

Transmit Octets Count Register – 0x80

This register is read only and cleared only during reset.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Transmit Octets Count	Counter for all transmited octets. Counter will wrap when limit is reached.

Table 5: Transmit Octets Count Register

Transmit Packet Count Register – 0x84

This register is read only and cleared only during reset.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Transmit Packet Count	Counter for all transmited packets. Counter will wrap when limit is reached.

Table 6: Transmit Packet Count Register

Receive Octets Count Register – 0x90

This register is read only and cleared only during reset.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Receive Octets Count	Counter for all valid received octets. Counter will wrap when limit is reached.

Table 7: Transmit Octets Count Register

Receive Packet Count Register – 0x94

This register is read only and cleared only during reset.

RESET VALUE: 0x00000000

Bit #	Name	Description
31 - 0	Receive Packet Count	Counter for all valid received packets. Counter will wrap when limit is reached.

Table 8: Transmit Packet Count Register

4. Clocks

This section describes the various clocks required by the 10GE MAC core.

Name	Frequency	Description
wb_clk_i	30 - 156Mhz	Wishbone interface clock.
clk_156m25	156.25Mhz	Clock for transmit and receive packet interfaces towards user's core logic. “pkt_tx_*” and “pkt_rx_*” signal are timed to this clock.
clk_xgmii_rx	156.25Mhz	Clock for XGMII receive interface between 10GE MAC core and XAUI/PCS macro.
clk_xgmii_tx	156.25Mhz	Clock for XGMII transmit interface between 10GE MAC core and XAUI/PCS macro.

Table 9: 10GE MAC Clocks

5. Resets

This section describes the various resets required by the 10GE MAC core. The user must ensure that each reset is synchronously de-asserted with it's corresponding clock. To ensure that transmit and receive FIFO's are initialized correctly, “reset_156m25_n”, “reset_xgmii_rx_n” and “reset_xgmii_tx_n” must be de-asserted within 2-cycles of each other.

Name	Description
wb_rst_i	Wishbone interface reset. Active high. Must be de-asserted synchronous to wb_clk_i.
reset_156m25_n	Core packet interfaces clock domain reset. Active low. Must be de-asserted synchronous to clk_156m25.
reset_xgmii_rx_n	XGMII receive clock domain reset. Active low. Must be de-asserted synchronous to clk_xgmii_rx.
reset_xgmii_tx_n	XGMII transmit clock domain reset. Active low. Must be de-asserted synchronous to clk_xgmii_tx.

Table 10: 10GE MAC Resets

6. IO Ports

This section specifies the 10GE MAC core IO ports.

Wishbone Interface

Refer to Wishbone interface specification for more details on these signals.

Port	Direction	Description
wb_adr_i [7:0]	Input	Address input. All accesses to core must be 32-bit. Bits 0 and 1 are not used.
wb_cyc_i	Input	Wishbone cycle.
wb_dat_i [31:0]	Input	Wishbone data input.
wb_stb_i	Input	Wishbone strobe.
wb_we_i	Input	Wishbone write enanle.
wb_ack_o	Output	Wishbone acknowledge.
wb_dat_o [31:0]	Output	Wishbone data output.
wb_int_o	Output	Wishbone interrupt signal. Active high.

Table 11: List of IO ports

Packet Receive Interface

This interface is used to transfer received packets to the FPGA/ASIC core logic.

Port	Direction	Description
pkt_rx_ren	Input	Receive Read Enable: This signal should only be asserted when a packet is available in the receive FIFO. When asserted, the 10GE MAC core will begin packet transfer on next cycle. Signal should remain asserted until EOP becomes valid.

Port	Direction	Description
pkt_rx_avail	Output	Receive Available: Indicates that a packet is available for reading in receive FIFO.
pkt_rx_data [63:0]	Output	Receive Data: Little-endian format. First byte of packet will appear on pkt_rx_data[7:0].
pkt_rx_eop	Output	Receive End of Packet: Asserted when the last word of a packet is read from receive FIFO.
pkt_rx_val	Output	Receive Valid: Indicates that valid data is present on the bus. This signal is typically asserted one cycle after pkt_rx_ren was asserted unless FIFO underflow occurs.
pkt_rx_sop	Output	Receive Start of Packet: Indicates that the first word of a frame is present on the bus.
pkt_rx_mod [2:0]	Output	Receive Packet Length Modulus: Valid during EOP. Indicates valid bytes during last word: Little Endian mode: 0: pkt_rx_data[63:0] is valid 1: pkt_rx_data[7:0] is valid 2: pkt_rx_data[15:0] is valid 3: pkt_rx_data[23:0] is valid 4: pkt_rx_data[31:0] is valid 5: pkt_rx_data[39:0] is valid 6: pkt_rx_data[47:0] is valid 7: pkt_rx_data[55:0] is valid Big Endian mode: 0: pkt_rx_data[63:0] is valid 1: pkt_rx_data[63:56] is valid 2: pkt_rx_data[63:48] is valid 3: pkt_rx_data[63:40] is valid 4: pkt_rx_data[63:32] is valid 5: pkt_rx_data[63:24] is valid 6: pkt_rx_data[63:16] is valid 7: pkt_rx_data[63:8] is valid
pkt_rx_err	Output	Receive Error: When asserted during a transfer, indicates that current packet is bad and should be discarded by user's logic. This signal is most likely

Port	Direction	Description
		asserted as the result of a CRC error.

Table 12: List of IO ports

Packet Transmit Interface

This interface accepts packets from the FPGA/ASIC core logic.

Port	Direction	Description
pkt_tx_data [63:0]	Input	Transmit Data: Little-endian format. First byte of packet must appear on pkt_rx_data[7:0].
pkt_tx_val	Input	Transmit Valid: This signal must be asserted for each valid data transfer to the 10GE MAC.
pkt_tx_sop	Input	Transmit Start of Packer: This signal must be asserted during the first word of a packet.
pkt_tx_eop	Input	Transmit End of Packet: This line must be asserted during the last word of a packet.
pkt_tx_mod [2:0]	Input	Transmit Packet Length Modulus: Valid during EOP. Indicates valid bytes during last word: Little Endian mode: 0: pkt_tx_data[63:0] is valid 1: pkt_tx_data[7:0] is valid 2: pkt_tx_data[15:0] is valid 3: pkt_tx_data[23:0] is valid 4: pkt_tx_data[31:0] is valid 5: pkt_tx_data[39:0] is valid 6: pkt_tx_data[47:0] is valid 7: pkt_tx_data[55:0] is valid Big Endian mode: 0: pkt_rx_data[63:0] is valid 1: pkt_rx_data[63:56] is valid 2: pkt_rx_data[63:48] is valid 3: pkt_rx_data[63:40] is valid 4: pkt_rx_data[63:32] is valid

Port	Direction	Description
		5: pkt_rx_data[63:24] is valid 6: pkt_rx_data[63:16] is valid 7: pkt_rx_data[63:8] is valid
pkt_tx_full	Output	Transmit Full: This signal indicates that transmit FIFO is nearing full and transfers should be suspended at the end of the current packet. Transfer of next packet can begin as soon as this signal is de-asserted.

Table 13: List of IO ports

XGMII Receive Interface

This interface accepts data or control character from XAUI/PCS macro. A normal XGMII interface is made of a 32-bit data bus clocked on rising and falling edges of the clock. For simplicity the 10GE MAC uses a 64-bit interface clocked on the rising edge only.

Port	Direction	Description
xgmii_rxc [7:0]	Input	XGMII Receive Control: Each bit corresponds to a byte on the 64-bit interface. When high, indicates that the byte is a control character. When low, indicates that the byte carries data.
xgmii_rxd [63:0]	Input	XGMII Receive Data: When interfacing with 32-bit devices, xgmii_rxd[31:0] should be mapped to the rising edge of the clock and xgmii_rxd[63:32] should be mapped to the falling edge.

Table 14: List of IO ports

XGMII Transmit Interface

This interface transmits data and control character towards the XAUI/PCS macro. A normal XGMII interface is made of a 32-bit data bus clocked on rising and falling edges of the clock. For simplicity the 10GE MAC uses a 64-bit interface clocked on the rising edge only.

Port	Direction	Description
xgmii_txc [7:0]	Output	XGMII Transmit Control: Each bit corresponds to a byte on the 64-bit interface. When high, indicates that the byte is a control character. When low, indicates that the byte carries data.
xgmii_txd [63:0]	Output	XGMII Transmit Data: When interfacing with 32-bit

Port	Direction	Description
		devices, xgmii_txd[31:0] should be mapped to the rising edge of the clock and xgmii_txd[63:32] should be mapped to the falling edge.

Table 15: List of IO ports

7. Waveforms

Wishbone Interface

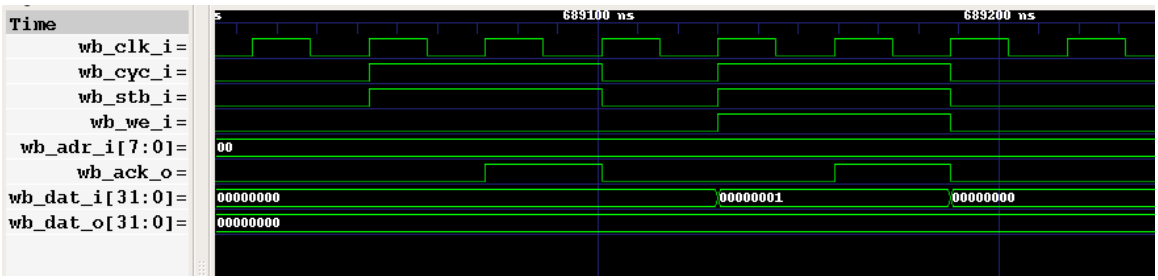

Figure 7: Wishbone Waveforms

Packet Receive Interface

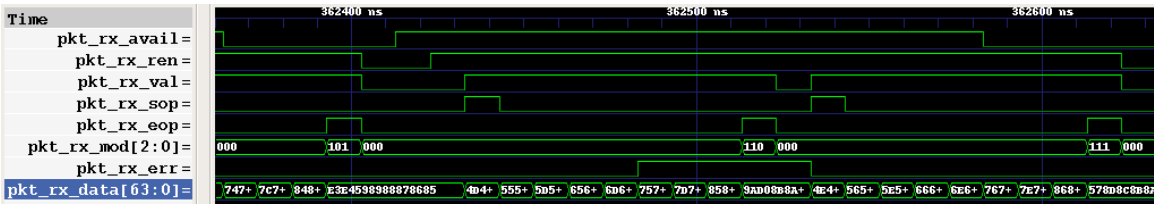

Figure 8: Packet Receive Interface Waveforms

Packet Transmit Interface

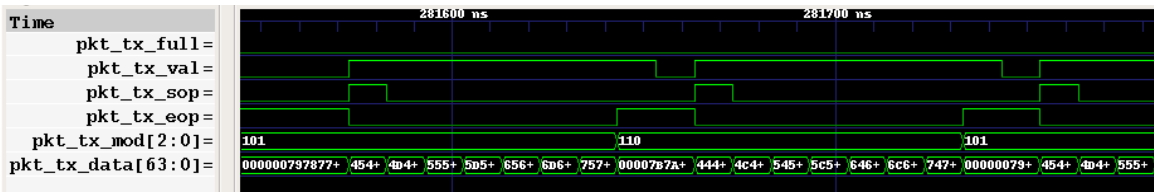

Figure 9: Packet Transmit Interface Waveforms

XGMII Receive Interface

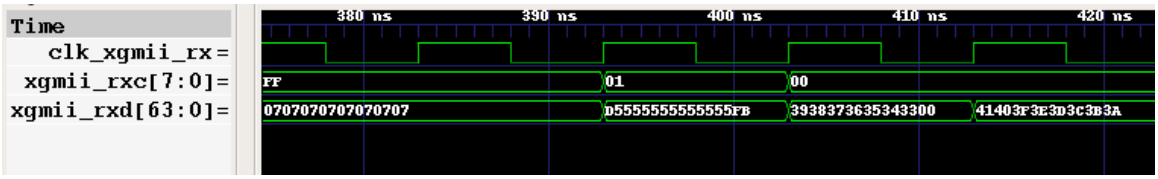

Figure 10: XGMII Receive Interface Waveforms

XGMII Transmit Interface

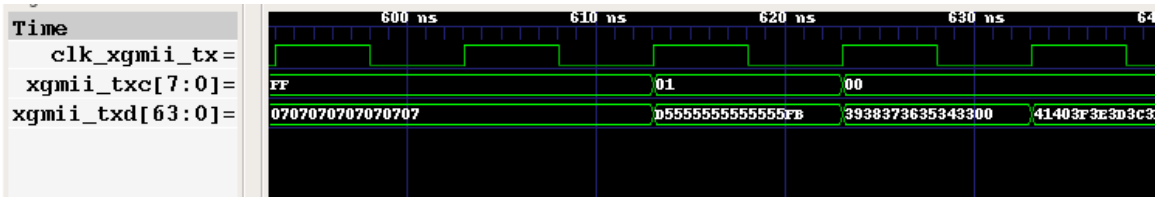

Figure 11: XGMII Transmit Interface Waveforms