LeWiz Ethernet MAC Controller

IP Core Specification

Core 3 – 100G/50G/40G/25G/10Gbps

LeWiz Communications, Inc.

"The Wizard of Internet Communications"

May 24, 2019
Revision 1.03

PO Box 9276
San Jose, CA 95157
www.lewiz.com
Email: support@lewiz.com

LeWiz can be contacted at: support@lewiz.com
or address: PO Box 9276, San Jose, CA 95157-9276
www.lewiz.com
Author: LeWiz Communications, Inc.

Change Log

Version	Significant Changes
1.03 OSrc	Updated for CORE3 support

Table of Contents

NOTE: This document is intended for SW/HW users using LeWiz MAC IP Core.

1 Introduction

LeWiz's MAC IP core (LMAC) is designed for use in a wide range of applications from small IoT devices to high performance computing systems. The core is intended for use on FPGA or in SoC ASICs. Its physical interface is designed to be general for fitting into a range of SerDes available in ASIC libraries or FPGA tools. When used as part of an SoC incorporated in a sensor for example, the LMAC acts as an Ethernet communication channel from the sensor to the main system or network. When such SoC is incorporated in a vehicle module (2nd example), the LMAC provides the connection from the vehicle module to the vehicle's Ethernet network. If incorporated into an FPGA (NIC card) and deployed in a data center system (such as a server or storage system, 3rd example), the LMAC provides the connection from the data center system to the data center network. LMAC Core3 is designed for very high performance to 100Gbps speed.

The LMAC architecture is designed to suit as the Ethernet interface of an end-point device or as a networking device such as a gateway or firewall. Ethernet is everywhere today. The LMAC core is useful to many things. The above are just a few examples.

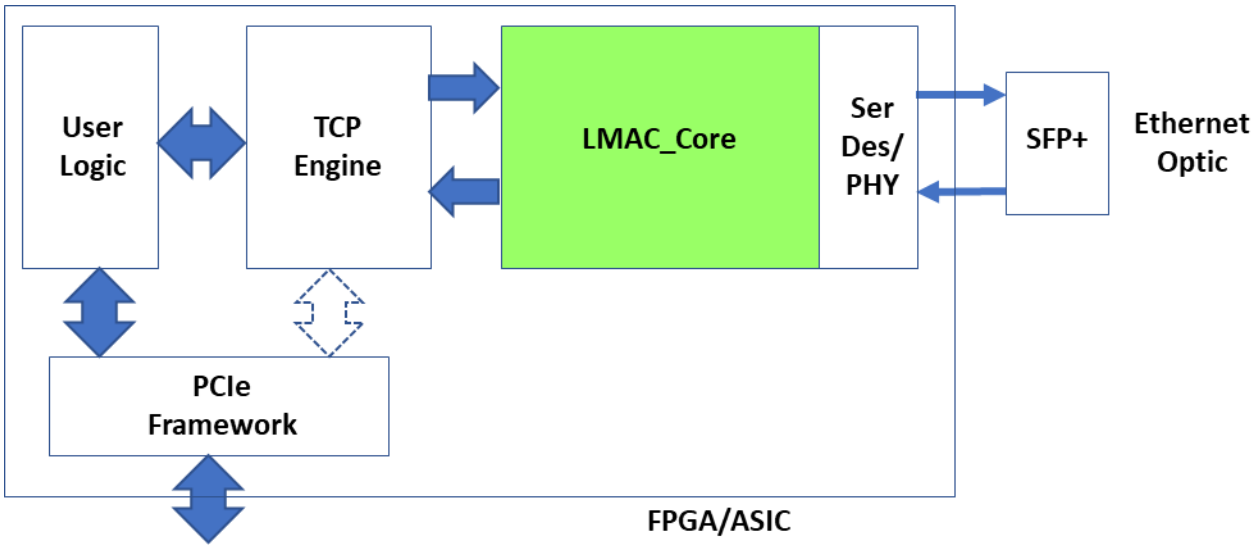


Figure 1.1: LMAC IP Core in a TCP/IP offload NIC

Figure 1.1 shows an example of the LMAC core in a TCP/IP offload engine (TOE) network interface card (NIC). It interfaces directly with the optical transceiver module (such as SFP+) on one side and the user logic on the other side. In a typical implementation, all of the modules shown above except the optical module can be contained in an FPGA device such as those available from Intel or Xilinx FPGA chip or an ASIC chip.

This specification provides the details for IP core user (such as chip designers) to incorporate the LMAC core into their chip and writing software for it. Where applicable,

this spec uses specific FPGA for example purposes. Other FPGA or ASIC device implementations would be similar.

The reader is expected to be knowledgeable with Ethernet, networking and FPGA/ASIC development.

(Where applicable LMAC Core 3 is described in here as much as possible. Other LMAC core information are provided for reference.)

1.1 Definitions, conventions and notations

- The word “connection” usually means TCP/IP connection.
- MAC = Media access controller (i.e. Ethernet controller chip in this case)
- B = in size, “B” means byte
- b = in size, “b” means bit
- lsb = least significant bit. Example: DATA[63:0], bit 0 is lsb, bit 63 is most significant bit (msb)
- API = Application programming interface
- CMD = command
- Reserved = unimplemented, undefined, or spare for future use or debug purpose.
- Optional = Optional are items that may or may not be supported or required.
- RX = Receive
- TX = Transmit
- Byte = 8 bit
- Word = 16 bit
- Double word = DWord = 32 bit
- Quad word = QWord = 64 bit
- SW = Software
- HW = Hardware
- Beat = each data slot (64 bit quantity and attributes) per clock pulse on the bus transfer is referred to as a beat.
- SIP = Source IP address
- DIP = Destination IP address
- Sport = Source TCP (or UDP) port
- Dport = Destination TCP (or UDP) port
- NIC = Network interface card
- PLL = Phase lock loop
- LVDS = low-voltage differential signaling
- SerDes = Serializer, De-serializer converting serial analog data into parallel digital data or vice versa.
- Ethernet Packet = Each Ethernet Packet has a packet header and a packet payload. The normal packet size is <1518 bytes total including the header.

- Ethernet Frame = An Ethernet frame is formed by encapsulating framing information around the Ethernet packet so the Ethernet packet is contained within the Ethernet frame. See the diagrams below.
- Pulse = a single clock assertion for the signal.
- User Logic = Logic blocks designed by the user which interfaces to the LMAC.
- PHY = Physical layer. For this document, it means the interface logic/analog circuit interfacing from the LMAC to the physical layer of the Ethernet. The physical layer can be copper or fiber optic. If the physical layer is copper, it may require some logic to interface from the LMAC PHY interface to an external copper PHY chip such as those from Marvel or others. If the physical layer is optical, it may require a SerDes to interface from the LMAC PHY interface to an external optical transceiver module. The specific design is dependent on specific physical medium and semiconductor technology and is beyond the scope of this spec.
- Promiscuous Mode = in Ethernet, Promiscuous is a mode where the LMAC accepts RX packets to any destination MAC address (even the destination MAC address does not match its own designated address). This mode may be used for debug or traffic capture purposes.

2 Overview

LMAC Core3 is designed for Ethernet speed 100G/50G/40G/25G/10Gbps. Figure 2.1 shows various main functions inside LeWiz MAC IP Core3.

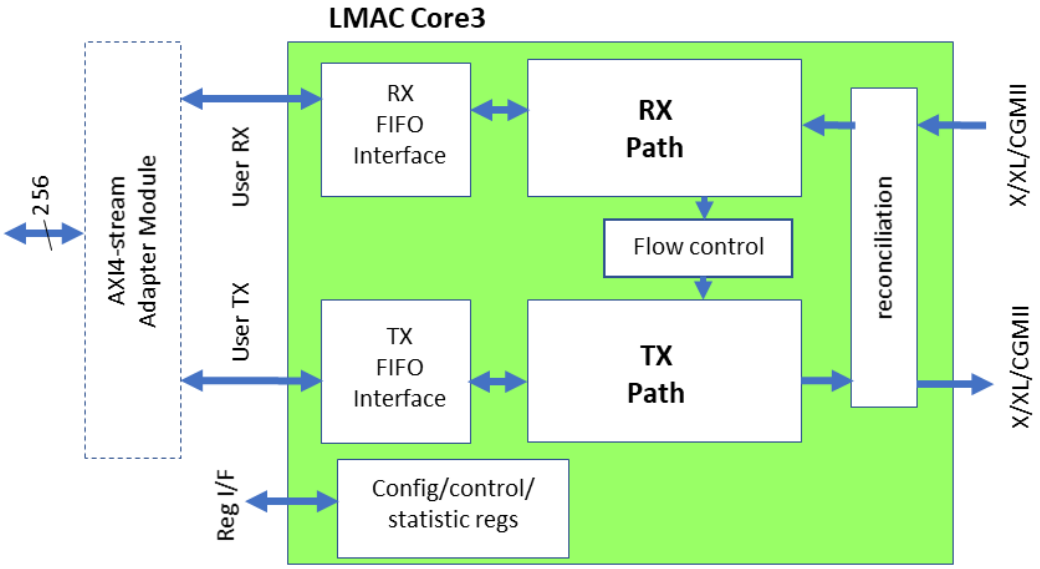


Figure 2.1: LMAC IP Core3 block diagram

The LMAC Core has 4 major blocks: the PHY interface (or reconciliation), the transmit path, the receive path, and register/configuration control. It contains internal FIFOs and provides a generic FIFO interface at User_TX/RX bus level. LeWiz also provides an optional AXI4-stream adapter module to interface from the FIFO interface (User_TX/RX) to an AXI4-stream bus. AXI4-stream bus is used by many chip manufacturers. Many FPGA architectures also support it. Either the AXI4 or FIFO interface can be used to connect to the user internal logic.

The LMAC Core design consists only of digital logic. It does not contain any analog component such as PLL (clock generation), SerDes, PHY, or LVDS I/O driver or the control unit associated with the analog component. These are complex functions by themselves. The LMAC source code is made available in Verilog RTL format.

The AXI4 adapter module provides the interface to the user's AXI4-stream bus. It also acts as a clock domain crossing interface and handles the complexity of the AXI bus for the LMAC's main core. User's internal clock may operate at high frequency such as GHz rate. The LMAC may operates only at 156.25MHz for 10Gbps Ethernet. In this case, the AXI4 adapter allows the different clock domain crossing. AXI4 also has various complexity such as inserting wait states in its data stream. The LMAC AXI4 has the buffer required to avoid holding off of wait states (reducing complexity in the user logic) and logic for handling AXI4 stream on the receive side similar to those methods found in MAC cores for FPGA devices.

The LMAC reconciliation block provides interface directly to the chip's SerDes. (The example shown in Figure 2.1 shows the X/XL/CGMII interface. XGMII is for 10Gbps. XLGMII is for 40Gbps. CGMII is for 100Gbps (4 SerDes lanes). The 25Gbps speed uses 1 lane of the CGMII interface. The 50Gbps speed uses 2 lanes of the CGMII interface.) The output of the chip's SerDes can drive the high-speed signals for interfacing to an external optical transceiver module (for example SFP+) without requiring an external PHY chip. The optical transceiver module can connect directly to the optical Ethernet network.

The LMAC transmit path on one side interfaces to the reconciliation block and generates all the signals and data format (including the CRC) required for Ethernet transmitting. It contains a deep FIFO to take packets from the user logic. On the user logic side, it uses the User TX Bus (described below) to allow the user logic to transmit packets out to the Ethernet network. Transmitting packets can be of different types: non-IP such as ARP, IP (such as PING), TCP or UDP data packets. For NIC application, transmitting of packets can be used for announcing IP addresses, establishing network node functions such as address tables, or transmitting actual data.

The receive block contains the logic required for the user logic to receive packets from the Ethernet network. It:

checks the CRC for data validity,
handles all Ethernet error conditions,
decapsulates the raw frame,
checks for valid MAC address
Supports Broadcast, multi-cast modes
supports VLAN (option), ARP, PING, TCP/UDP, IP only, etc.
Supports frames of different sizes (option),
collects statistics for analysis and debug

The register control block contains the logic required for tracking the statistics and physical link information. These are information required by the operating system or user to use in debug and network analysis. It interfaces with the user logic via its own lower performance interface.

The interfaces to the user logic and the registers are described in later sections.

3. Description of Transmit path inside LMAC Core logic

3.1. TX Path Overview

In the transmit path, the TX FIFO stores packets from the user until the packet can be transmitted. Data is written into the TX FIFO from User TX bus. (If the AXI4 bus is used the transmit data originated from the AXI4-stream TX interface.) A whole Ethernet packet must be sent in TX path for it to transmit. The TX path uses the Ethernet packet information to build the Ethernet frame for transmitting. The Ethernet packet contains the Ethernet layer, IP layer, TCP layer, payload and any padding which may be required (see Figure 3.1.1). Padding is used to increase the size of small packets to meet the minimum 64-byte Ethernet frame requirement.

The width of the user interface side is as follows:

Speed	User Interface
100G/50G/40G/25G/10Gbps	256bit

(The user interface's clock speed can be independent of the core's internal clock speed.)

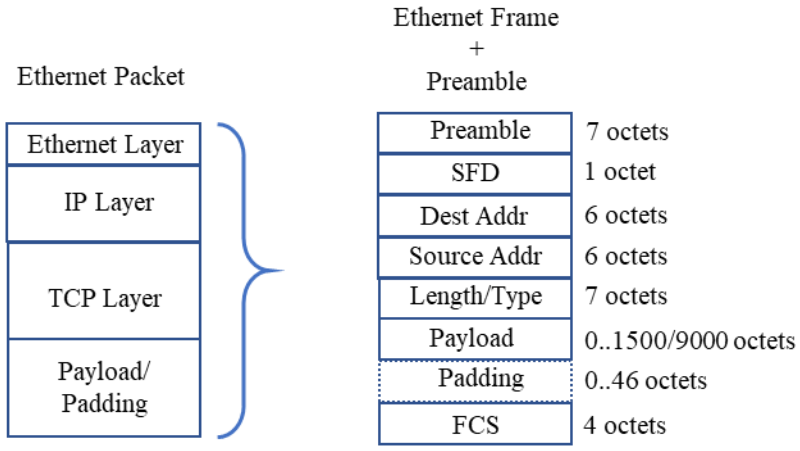


Figure 3.1.1: Ethernet packet and frame formats

On the Ethernet physical side, the TX path provides different physical interfaces. Different Ethernet speed, different medium (optic or copper) requires different PHY interface.

The PHY support is as follows:

Core Name	Speed	PHY Interfaces	Notes

Core3	100G/50G/40G/25G/10Gbps	X/XL/CGMII	156.25MHz for XGMII 156.25MHz for XLGMII 390.625MHz for CGMII (100G/50G/25G)

XGMII requires 1 SerDes lane (10Gbps)
XLGMII requires 4 SerDes lanes (backward compatible with 40Gbps)
CGMII requires 4 SerDes lanes (each lane 25Gbps; total 100Gbps)
25Gbps Ethernet requires 1 SerDes lanes
50Gbps Ethernet requires 2 SerDes lanes
Although the core's physical interface may vary in width, the user interface for Core3 is always 256 bits providing 1 consistent interface to the user logic.

The configuration pin interface of the LMAC Core3 provides signals that the user can directly tie to 1 or 0 to select the desired LMAC speed. See Section 6.2.2.

3.2. CRC

CRC is generated by the LMAC on the transmit path for the FCS field of the Ethernet frame.

For the receive path, if enabled the CRC-32 field is always checked and flagged if error is detected. Or it can be ignored if CRC check is disabled by software using the control register. The CRC polynomial, as specified in the IEEE 802.3 standard, is:

FCS(X) = X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 + X8 + X7 + X5 + X4 + X2 + X1 + 1

Some applications such as network analyzer would need to be able to receive bad CRC packets as well as good packets. Software can also enable the Core to forward bad CRC packets to the FIFO interface. CRC field is removed from the packet before the packet is sent to the FIFO interface for the user logic.

3.3. TX FIFO Interface - Signal Description

(AXI4-stream signals are described in the AXI4-stream chapter below)

All interfaces for LeWiz MAC Core are synchronous and operate based on rising edge of the clock. This is for easy timing closure in FPGA or SoC designs.

The transmit bus (User TX) is a FIFO interface. Its information are as follows:

Notations:
- U→ Means from user logic to LeWiz MAC
- ←M Means from LeWiz MAC to user logic
- ←→ Means bidirectional

Signal timing

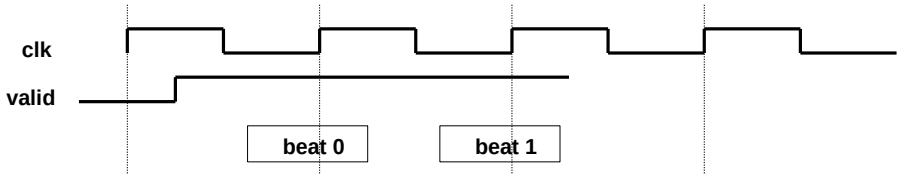


Figure 3.1: Valid signal timing

Signal	Direction	Description	
Clk	U→	Clock Activity can only occur on the rising edge of the clock.	
Transmit Bus			
TX_WE	U→	Write enable. 1 = data is valid on the TX_DATA bus. 0 = no data is on the data bus	
TX_DATA[255:0]	U→	256-bit data bus The first qqword sent by the user logic is the byte count of the packet as follows: Qwd0[15:0] = packet byte count (total number of bytes in the header and payload of the packet. Example 16'h003C for smallest 64 byte Ethernet packet.) Subsequent qwords after the packet byte count are the Ethernet packet header and the payload. The format of the packet follows the Ethernet packet format which is similar to what is shown by standard tools such as Wireshark. The data on this TX_DATA however must be in big-endian for the packet data. (See example of ARP packet below)	
TX_BE[31:0]	U→	(optional) Byte enable of the 256-bit data bus. 1 bit for each byte lane of the data bus. TX_BE[0] = if 1, means data is valid on TX_DATA[7:0] And TX_BE[7] = if 1, means data is valid on TX_DATA[63:56] And so on The internal logic of the LMAC generates its own byte enable.	
TXFIFO_FULL	←M	1 = the internal TX FIFO is full and cannot accept any more packet.	

TXFIFO_WUSED_QWD[12:0]	←M	Indicating the number of qwords the TX FIFO contained. This value is dynamic and can change from clock to clock. (Current implementation only uses bits [9:0] for FIFO size 1Kx64)	

Example of a Gratuitous ARP Reply Ethernet packet on TX_Data[63:0] bus:
(bit 0 is the right most bit)

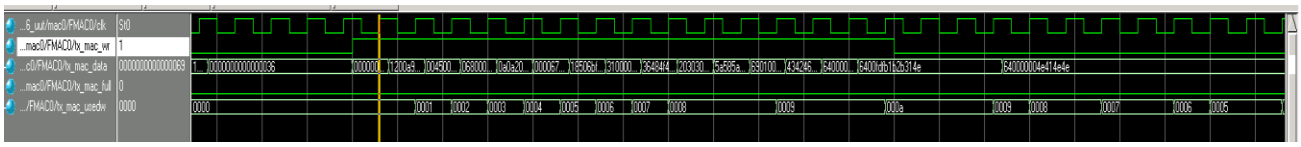
```
# tx_data:      0000_0000_0000_003c      //3C = 60 bytes (BCNT)
# tx_data:      1200_FFFF_FFFF_FFFF
# tx_data:      0100_0608_f0ff_ee32
# tx_data:      1200_0200_0406_0008
# tx_data:      200a_0a0a_f0ff_ee32
# tx_data:      0a0a_f0ff_ee32_1200
# tx_data:      0000_0000_0000_200a
# tx_data:      0000_0000_0000_0000
# tx_data:      0000_0000_0000_0000
```

For transmitting, the packet must conform to Ethernet standard and minimum packet size is 64 byte. (The CRC is 4 bytes)

If LeWiz AXI4 adapter module is used in the design, the AXI4 adapter uses the user's Ethernet packet information and generates the FIFO interface format with the BCNT field automatically for the LMAC.

3.4. TX FIFO Interface - Timing Diagram

The following figure shows the relationship of the signals in the transmit bus.



4. Description of Receive path inside Core logic

This is the main block for processing the received data and directing it to different outputs for the user logic. The receive path takes care of all the peculiarities of Ethernet and always output aligned Ethernet packets to the user logic. The RX path detects the MAC address, extracts the Ethernet packet from the Ethernet frame, checks the CRC. Specifically, the RX path:

- Receives Ethernet frames from the PHY layer, aligns the frame to the width of the Core (different widths for different Ethernet speeds)
- Extracts the Ethernet packet from the frame and sends the packet on to the user logic side
- Checks Ethernet frames for mal-form frames

- Removes the frame preamble and frame SFD fields
- Filters out packets not matching Destination MAC Address field (if not in promiscuous mode)
- Processes Pause and PFC Frames (option)
- Checks the Ethernet frame Length
- Calculates and verifies CRC-32 (Ethernet FCS)
- Writes the received Frames in the Core Receive FIFO for the user logic.

The user logic has 2 options for retrieving the RX packets from the LMAC. One is to use the RX FIFO interface, i.e. read the packets out from the RX FIFO (describing next). Another option is to interface to the AXI4-stream interface (described in the AXI4-stream chapter).

4.1 RX FIFO Interface - Signal Description

These are signals at the User_RX bus (also referred to as FIB or front-end MAC internal bus). The RX Path contains RX FIFOs which can be read from to obtain packet(s) and the packet's byte count (or size of the packet).

After a packet has been received and verified, the LMAC puts the packet's data and the packet's byte count into 2 RX FIFOs – a packet byte count FIFO and a packet data FIFO. Some applications do require a packet's byte count before the packet can be processed. This mechanism provides access to both. The interface description below describes the interfaces for reading the packet's byte count and the packet's data.

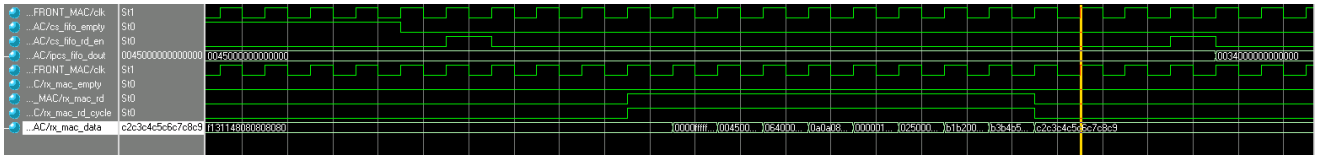
Signal	Direction	Description	
Clk	U→	Clock Activity can only occur on the rising edge of the clock. (Or share the clock with the TX path)	
Receiving Bus – packet byte count/size interface (IPCS)			
RX_IPCS_EMPTY	←M	Byte count FIFO's empty signal. 0 = a byte count is available for reading. 1 = no byte count is available	
RX_IPCS_RDEN	U→	Read enable for the byte count FIFO. User asserts 1 pulse to read 1 qword from the byte count FIFO for the size of 1 packet.	
RX_IPCS_DATA[63:0]	←M	64-bit IPCS data bus The IPCS bus is only intended for reading 1 qword at a time. Currently: [63:48] = packet's byte count [47:0] = reserved	
Receiving Bus – packet data interface (after reading the byte count the user can read out the packet's data – 1 entire packet at a time)			

RX_PKT_EMPTY	←M	Packet data FIFO's empty signal. 0 = a packet is available for reading. 1 = no packet is available	
RX_PKT_RDEN	U→	Read enable for the packet data FIFO. User asserts 1 pulse to read 1 qword from the packet data FIFO.	
RX_PKT_RD_CYCLE	U→	Indicates a user packet read cycle is in progress. This signal is used to indicate to the LMAC that a user packet read cycle is in progress which can be lengthy depending on the size of the packet. Asserts high to start the cycle and only negates when the user's cycle is fully complete.	
RX_PKT_DATA[255:0]	←M	Packet's data bus. For each RDEN a qword is sent out to the user from the packet data FIFO. The full qword is always provided. Based on the packet's byte count the user can determines how many bytes is available or remaining to be read from the packet. Similar to FPGA FIFO, it takes 1 clock for data to be available from the read enable.	

The timing diagram below illustrates both RX buses for packet byte count and packet data.

4.2. RX FIFO Interface - Timing

The top signals are for packet byte count reading.
The bottom signals (separated by the *clk* signal) are for packet data reading.



5. AXI4-Stream Interface

The AXI4-stream interface is designed to be similar to those available for FPGA devices. The user must incorporate LeWiz optional AXI4 adapter module to obtain this interface. The AXI4 adapter module can be connected to the LMAC's TX/RX FIFO interface directly. The LMAC test bench shows how this connection is made.

The AXI4-stream bus divides into a TX part and an RX part. These are described below.

The AXI4-stream interface can operate at a different clock frequency than the LMAC clock.

5.1 AXI4-stream TX Interface - Signal Description

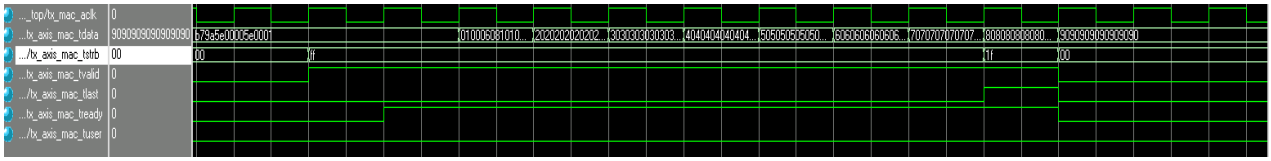
TX Signals for AXI4-stream begins with TX_AXIS_*

(Main bus signals are described here to illustrate the functions.)

Signal	Direction	Description	
AXIS_Clk	U→	User AXI bus clock Activity can only occur on the rising edge of the clock. This clock can be different speed than the LMAC clock.	
		Main TX Bus	
TX_AXIS_TDATA[N:0]	U→	TX Stream Data bus N = 255 for Core 3	
TX_AXIS_TSTRB[K:0]	U→	TX bus strobe. Drives out along with the TDATA bus. 1 strobe bit for each 8-bit lane. If the strobe bit is 1, it means data is available on the lane. Strobe 0 = lane 0 = [7:0] Strobe 1 = lane 1 = [15:8] Strobe 2 = lane 2 = [23:16] Etc.	
TX_AXIS_TVALID	U→	TX bus valid Must be 1 for each valid data beat on the data bus. If not valid (i.e. 0), both the TSTRB and TDATA must be ignored.	
TX_AXIS_TLAST	U→	TX bus last beat indicator Must be 1 for the last valid data beat on the data bus.	
TX_AXIS_TREADY	←M	TX bus is ready to receive the data When this is 1, the LMAC is ready to receive a full packet of 1500 data bytes. If 0, it means the LMAC FIFO is full and cannot receive more transmit data.	
TX_AXIS_TUSER	U→	TX bus error indicator. Reserved signal. (Not used. User logic should qualify its data before sending to the LMAC and should not send bad data into the LMAC.)	

5.2 AXI4-stream TX Interface – Timing

Timing of the AXI4 stream TX signals.

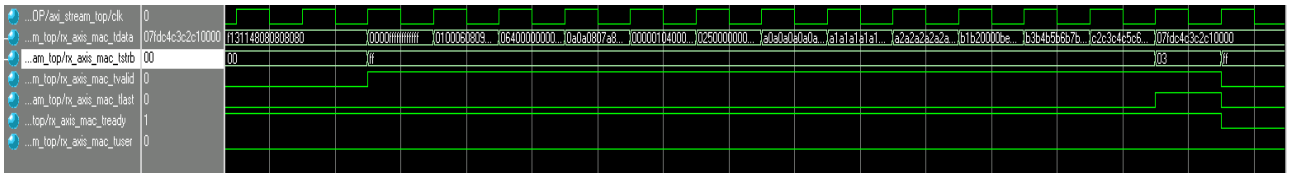


5.3 AXI4-stream RX Interface – Signal Description

Signal	Direction	Description	
AXIS_Clk	U→	User AXI bus clock Activity can only occur on the rising edge of the clock. This clock can be different speed than the LMAC clock. The TX and RX stream interfaces share a clock.	
		Main TX Bus	
RX_AXIS_TDATA[N:0]	←M	RX Stream Data bus N = 255 for Core 3	
RX_AXIS_TSTRB[K:0]	←M	RX bus strobe. Drives out along with the TDATA bus. 1 strobe bit for each 8-bit lane. If the strobe bit is 1, it means data is available on the lane. Strobe 0 = lane 0 = [7:0] Strobe 1 = lane 1 = [15:8] Strobe 2 = lane 2 = [23:16] Etc.	
RX_AXIS_TVALID	←M	RX bus valid Must be 1 for each valid data beat on the data bus. If not valid (i.e. 0), both the TSTRB and TDATA must be ignored.	
RX_AXIS_TLAST	←M	RX bus last data beat indicator Must be 1 for the last valid data beat on the TDATA bus.	
RX_AXIS_TREADY	U→	User RX bus is ready to receive the data (This signal is used in conjunction with the COMPATIBLE_MODE side band signal on the LMAC. If COMPATIBLE_MODE = 1, the LMAC always assume the user logic is ready to receive the data. This is similar to MAC controllers from FPGA devices. COMPATIBLE_MODE = 1 is the only mode supported. Wait state between the TDATA beats during the data transfer is not supported.)	
RX_AXIS_TUSER	←M	RX bus error indicator. Used by the LMAC to indicate error condition has been detected on the received Ethernet packet.	

5.4 AXI4-stream RX Interface – Timing

Timing of the AXI4 stream RX signals



6. LMAC Register Interface, Configuration, Clock and Reset

This register interface is mainly used to read the internal registers of the LMAC core. These are status and statistic information collected by the hardware.

To configure the LMAC Core a set of external configuration signals are also provided (see below in the configuration section). The LMAC configuration information are implemented as signals so the user logic has the flexibility of configuring the core through its own registers or via direct logic such as state machines.

The LMAC also uses various clocks and resets. These are also described below.

6.1 Clocks and Reset Signals

These are the clocks and resets used by the LMAC more to be supplied by the user logic or by the board level logic which is implementation dependent.

Signal	Direction	Description	
Clk	U→	This is the main clock for the LMAC core. Examples of the speeds are: 156MHz for 10Gbps. Activity can only occur on the rising edge of the clock.	
Reg_Clk	U→	Register interface clock For speed 156MHz and below this can be tied to the main Clk signal.	
Reset_	U→	General reset signal. Digital reset. Low true. This is also the soft reset.	
Phy_Reset_	U→	PHY Reset. Same as the reset for user's SerDes, PHY device(s) or other analog related modules/chips. Low true. This may also be the board level reset or hard reset or power on reset. Phy_Reset_ must occur first before logic Reset_ Recommendation: Allows 3mS between the 2 events. This is user system implementation dependent, however.	

6.2 Configuration Signals

The following are static signals used to configure the LMAC Core.

Signal	Direction	Description	
Linkup	←M	Static signal from the LMAC. If 1, indicating the Ethernet link is up.	
MAC_CTRL[31:0]	U→	32 bit control signals (See MAC_CTRL below)	
MAC_CTRL1[31:0]	U→	Second set of 32-bit control signals (See MAC_CTRL1 below)	
MAC_RXD_EN	U→	LMAC receive path enable. This is a static signal. It allows the user logic to control the receiving function of the LMAC. It's useful in debug support. 1 = RX enable	

6.2.1 MAC_CTRL Configuration Detail

Default values in the table below are values recommended for NIC applications.
Note that reserved fields may be used internally (or future functions) by the LMAC so they should be set as shown below.

Each of the LMAC register's default values enable it to operate after power on without requiring software to configure it.

Bit #	Field name	Default value	Description
0	TX_XO_EN	1'b0	Enable the TX Path to transmit PAUSE frame for stopping other sources from sending any more packet(s) to this port (XOFF event) or transmitting XON frame. 0 = disable PAUSE frame transmission 1 = enable PAUSE frame transmission
1	RX_XO_EN (1 bit)	1'b0	Enable the RX Path to receive PAUSE frame. 0 = disable PAUSE frame receipt 1 = enable PAUSE frame receipt
2	Reserved	1'b0	
3	RX_CRC_EN	1'b0	Enable CRC checking 0 = disable 1 = enable

			If enabled, incoming CRC will be checked. Bad CRC packet will be dropped. If disabled, checking of bad CRC is disabled and bad CRC packets will be forwarded as if they are good CRC packets. This is mainly for diagnostic purposes.
4	PROMIS_MODE_EN	1'b1	Enable Promiscuous mode (see definition) 0 = disable 1 = enable
5	Reserved	1'b0	
6	TX_CNT_AUTO_CLR_EN	1'b0	If set to 1, enable the auto clearing of TX statistic registers upon a read to the register is detected. Only a few registers has this capability. 0 = disable 1 = enable
7	RX_CNT_AUTO_CLR_EN	1'b0	If set to 1, enable the auto clearing of RX statistic registers upon a read to the register is detected. Only a few registers has this capability. 0 = disable 1 = enable
[8:9]	Reseved	2'b00	
[10]	MCAST_EN	1'b1	Enable Receiving Multicast packets. 1'b0 = disable 1'b1 = enable
[11]	ACCEPT_BRDCST_EN	1'b1	If enabled, LMAC will accept received broadcast packets. 1'b0 = disable 1'b1 = enable
[12]	Reserved	1'b0	
13	Reserved	1'b0	
14	Reserved	1'b0	
15	Reserved	1'b0	
19:16	Reserved	4'h0	
23:20	Reserved	4'h0	
31:24	Reserved	8'h0	

6.2.2 MAC_CTRL1

Second set of configuration control signals

Bit #	Field name	Reset value	Description
31:28	Reserved	4'h0	
27	Reserved		
26:24	Reserved	3'h0	
23	Reserved		
22:20	Reserved	3'b000	
19	Reserved	1'b0	
18	Reserved	1'b0	

17:16	Reserved	2'b00	
15	Reserved	1'b0	
14	Reserved	1'b0	
13:0	MAX_PKT_SIZE (14 bits)	14'h5EE	This field configures the max packet size in byte for the Ethernet port. Typically, Ethernet packets are between 64 and 1518 (5EE hex) bytes. This value is applicable for most applications.

In addition, LMAC Core3 top level has 3 speed setting bits:
(NOTE that these signals are static and MUST NOT change during normal operation.)

To avoid user from falsely setting the bits, in AXIS_LMAC_TOP, the design example uses a single bus signal FMAC_SPEED[2:0] to set the speed of the LMAC. This example code defined the speeds as follow:
LMAC_SPEED[2:0] = 000 = 10Gbps;
LMAC_SPEED[2:0] = 001 = 25Gbps;
LMAC_SPEED[2:0] = 010 = 40Gbps;
LMAC_SPEED[2:0] = 011 = 50Gbps;
LMAC_SPEED[2:0] = 100 = 100Gbps;
Other values are reserved

6.3 Register Interface

This section describes the signals required for accessing the internal registers of the LMAC and the available register set.

6.3.1 Register Interface Signals

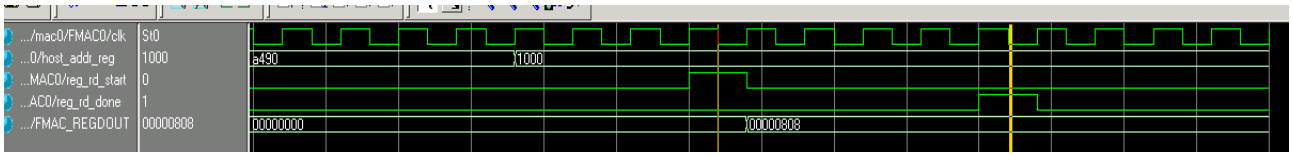
These signals are operating on user logic clock (156MHz or below). Its function is similar to the standard Advanced Peripheral Bus.

Signal	**Direction**	**Description**	
Reg_Clk	U→	Register Clock (or user logic clock for example 156MHz) Activity can only occur on the rising edge of the clock.	
		Host Address Bus	
host_addr[15:0]	U→	16 bit host byte address bus for selecting a memory mapped register. Valid 1 clk before register read start signal. Address must be 64-bit aligned for most cases.	
reg_rd_start	U→	Pulse. Register read start. 1 = Start the read for a register in the LMAC core 0 = idle.	
reg_rd_done_out	←M	Pulse. Indicating the register read is done and its data is available on the MAC_REGDOUT bus.	

		1 = indicating the data is available for the read to register 0 = data not available. This signal is the 5-clock delay of the register read start signal.	
Read Completion Data Bus			
mac_regdout[31:0]	←M	Data returned from LMAC register. Read is 32 bit at a time.	

6.3.2 Register Interface Timing

(For illustration only)



6.3.3 Available Registers

The available registers are shown in the table below.
Addr_offset is the same as the host_addr bus above.

Most of these are counters of specific condition captured by the LMAC. The FMAC_PHY_STAT register is described below.

Addr Offset	Reg Name	Description
h'0_1000	Reserved	(Reserved - user defined space)
h'0_1008	Reserved	
h'0_1010	Reserved	
h'0_1018	Reserved	
h'0_1020	FMAC_TX_PKT_CNT	Number of transmitted packets. (Each register is 32 bit)
h'0_1028	FMAC_RX_PKT_CNT_LO	Number of received packets (bit 31:0 of 64 bit counter)
h'0_102C	FMAC_RX_PKT_CNT_HI	Number of received packets (bit 63:32 of 64 bit counter)
h'0_1030	FMAC_TX_BYTE_CNT	Number of bytes transmitted
h'0_1038	FMAC_RX_BYTE_CNT_LO	Number of bytes received (low part of 64 bit count)
h'0_103C	FMAC_RX_BYTE_CNT_HI	Number of bytes received (high part of 64 bit count)
h'0_1040	FMAC_RX_UNDERSIZE_PKT_CNT	Number of undersize packets received (<64 byte packet)
h'0_1048	FMAC_RX_CRC32_ERR_CNT	Number of CRC error packets encountered
h'0_1050	FMAC_RX_DCNT_OVERRUN	Number of packets overrun the RxFIFO and dropped
h'0_1058	FMAC_RX_DCNT_LINK_ERR	Number of packets received encountered link error
h'0_1060	FMAC_RX_PKT_CNT_OVERSIZE	Number of packets received but over the MAX packet size
h'0_1068	FMAC_PHY_STAT	Internal PHY/Ethernet Link status and information
h'0_1070	Reserved	
h'0_1078	FMAC_RX_PKT_CNT_JABBER	Number of jabber packets
h'0_1080	FMAC_RX_PKT_CNT_FRAGMENT	Number of fragmented packets
h'0_1088	FMAC_RX_RAW_FRAME_CNT	Number of raw Ethernet frames received

h'0_1090	FMAC_RX_BAD_FRAME_CNT	Number of bad Ethernet frames received
h'0_1800	FMAC_RX_PKT_CNT64_LO	Number of packets with size ≤ 64 bytes (low 32 bit count)
h'0_1804	FMAC_RX_PKT_CNT64_HI	Number of packets with size ≤ 64 bytes (high 32 bit count)
h'0_1808	FMAC_RX_PKT_CNT127_LO	Number of packets with size ≤ 127 bytes (low 32 bit count)
h'0_180C	FMAC_RX_PKT_CNT127_HI	Number of packets with size ≤ 127 bytes (high 32 bit count)
h'0_1810	FMAC_RX_PKT_CNT255_LO	…….
h'0_1814	FMAC_RX_PKT_CNT255_HI	…….
h'0_1818	FMAC_RX_PKT_CNT511_LO	…….
h'0_181C	FMAC_RX_PKT_CNT511_HI	…….
h'0_1820	FMAC_RX_PKT_CNT1023_LO	…….
h'0_1824	FMAC_RX_PKT_CNT1023_HI	…….
h'0_1828	FMAC_RX_PKT_CNT1518_LO	Number of packets with size ≤ 1518 bytes (low 32 bit count)
h'0_182C	FMAC_RX_PKT_CNT1518_HI	Number of packets with size ≤ 1518 bytes (high 32 bit count)
h'0_1830	Reserved	
h'0_1834	Reserved	
h'0_1838	Reserved	
h'0_183C	Reserved	
h'0_1840	Reserved	
h'0_1844	Reserved	
h'0_1848	Reserved	
h'0_184C	Reserved	
h'0_1850	Reserved	
h'0_1854	Reserved	
h'0_1858	Reserved	
h'0_185C	Reserved	

Each register is 32 bit width. 64 bit registers are indicated as "lo" and "hi"

6.3.4 FMAC_PHY_STAT Register

FMAC_PHY_STAT[0] = if 1, indicating the Ethernet link is up
FMAC_PHY_STAT[1] = Reserved
FMAC_PHY_STAT[2] = if 1, indicating the RX of the SerDes channel is aligned

Other bits are reserved.

7. PHY Interface Signals

For this document, the term PHY means the interface logic/analog circuit interfacing from the LMAC to the physical layer of the Ethernet. The Ethernet physical layer can be copper or fiber optic. If the physical layer is copper (as in 10Gbps speed case), it may require logic circuits to interface from the LMAC XGMII interface to an external copper PHY chip. If the physical layer is optical, it may require a SerDes to interface from the LMAC XGMII (for example) to an external optical transceiver module.

Different PHY interface is provided by the LMAC for different Ethernet speed, as follows:

Core Name	Speed	PHY Interfaces	Notes
Core3	100G/50G/40G/25G/10Gbps	X/XL/CGMII	

The PHY interfaces are Ethernet standard. The subsequent sections describe the signal implementation for each interface.

7.1 XGMII Interface

XGMII is a 64bit, 10Gbps interface standard based on 1 SerDes lane. Its signals are as follows:

Signal	Direction	Description	
		Transmit signals	
XGMII_TxD[63:0]	M→PHY	Transmit data bus (Operates at 156.25MHz)	
XGMII_TxC[7:0]	M→PHY	Transmit control bus. Each control bit is corresponding to an 8bit data lanes. TxC[0] = controls data lane 0 or TxD[7:0] TxC[1] = controls data lane 1 or TxD[15:8] TxC[2] = controls data lane 2 or TxD[23:16] TxC[3] = controls data lane 3 or TxD[31:24] TxC[4] = controls data lane 4 or TxD[39:32] TxC[5] = controls data lane 5 or TxD[47:40] TxC[6] = controls data lane 6 or TxD[55:48] TxC[7] = controls data lane 7 or TxD[63:56] If TxC[n] equals to 1, it indicates to the PHY that information available on the corresponding XGMII_TxD lane is a control byte. If 0, it means the information is a data byte. XGMII Control byte indicates IDLE, SOF, EOF, or ERROR conditions present on the data lane.	
		Receive signals	
XGMII_RX_CLK	M←PHY	RX Clock from PHY to LMAC (156.25Mhz)	
XGMII_RxD[63:0]	M←PHY	Received data bus	
XGMII_RxC[7:0]	M←PHY	Each control bit is corresponding to an 8bit data	

		lanes. RxC[0] = controls data lane 0 or RxD[7:0] RxC[1] = controls data lane 1 or RxD[15:8] RxC[2] = controls data lane 2 or RxD[23:16] RxC[3] = controls data lane 3 or RxD[31:24] RxC[4] = controls data lane 4 or RxD[39:32] RxC[5] = controls data lane 5 or RxD[47:40] RxC[6] = controls data lane 6 or RxD[55:48] RxC[7] = controls data lane 7 or RxD[63:56] If RxC[n] equals to 1, it indicates to the LMAC that information available on the corresponding XGMII_RxD lane is a control byte. If 0, it means the information is a data byte. XGMII Control byte indicates IDLE, SOF, EOF, or ERROR conditions present on the data lane.	

7.2 Other PHY Interfaces

Other higher speed PHY interfaces (such as XLGMII) are formed by adding more SerDes lanes to obtain larger busses. For example, 40Gbps XLGMII interface has 4 SerDes and is 4 times larger in width than the XGMII interface. CGMII interface also has 4 SerDes but operates at higher clock rate 390.625MHz to obtain 100Gbps bit rate.