Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

10 Gb+ Ethernet MAC IP Core User's Guide

December 2010
IPUG39_02.9

Table of Contents

Chapter 1. Introduction 4
Quick Facts 4
Features 4
Chapter 2. Functional Description 6
Receive MAC 8
Transmit MAC 10
Signal Descriptions 11
Timing Specifications 13
Transmit Interface 13
Receive Interface 13
Chapter 3. Parameter Settings 14
Parameter Descriptions 14
Multicast Address Filter 14
Evaluation Generation Options 15
Chapter 4. IP Core Generation 16
Licensing the IP Core 16
Getting Started 16
IPexpress-Created Files and Top Level Directory Structure 18
Instantiating the Core 20
Running Functional Simulation 20
Synthesizing and Implementing the Core in a Top-Level Design 21
Hardware Evaluation 23
Enabling Hardware Evaluation in Diamond 23
Enabling Hardware Evaluation in ispLEVER 23
Updating/Regenerating the IP Core 23
Regenerating an IP Core in Diamond 23
Regenerating an IP Core in ispLEVER 24
Chapter 5. Application Support 25
Reference Register Descriptions 25
Chapter 6. Support Resources 45
Lattice Technical Support 45
Online Forums 45
Telephone Support Hotline 45
E-mail Support 45
Local Support 45
Internet 45
References 45
LatticeECP2/M 45
LatticeECP3 45
LatticeSC/M 45
Revision History 46
Appendix A. Resource Utilization 47
LatticeECP2 and LatticeECP2S FPGAs 47
Ordering Part Number 47
LatticeECP2M and LatticeECP2MS FPGAs 47
Ordering Part Number 47
LatticeECP3 FPGAs 48
Ordering Part Number 48
LatticeSC/M FPGAs 48

Ordering Part Number 48

Introduction

This document provides technical information about the Lattice 10 Gigabit Plus (10 Gb+) Ethernet Media Access Controller (MAC) Intellectual Property (IP) core. The 10 Gb+ Ethernet MAC IP core comes with the following documentation and files:

- Protected netlist/database
- Behavioral RTL simulation model
- Source files for instantiating and evaluating the core

Quick Facts

Table 1-1 gives quick facts about the 10 Gb+ Ethernet MAC IP core for LattceECP2™, LatticeECP2M™, LatticeECP3™, LatticeSC™, and LatticeSCM™ devices.

Table 1-1. 10 Gb+ Ethernet MAC IP Core Quick Facts

		10 Gb+ Ethernet MAC IP Configuration				
Core Requirements	FPGA Families Supported	LatticeECP2	LatticeECP2M	LatticeECP3	LatticeSC	LatticeSCM
	Minimal Device Needed	LFE2-35E-7F672C	LFE2M35E-7F672C	LFE3-35EA-8F672CES	LFSC3GA25E-6F900C	LFSCM3GA25EP1-6F900C
Resources Utilization	Target Device	LFE2-35E-7F672C	LFSCM3GA25EP1-5F900C	LFE2M35E-7F 672C	LFSC3GA25E-6F900C	LFSCM3GA25EP1-6F900C
	Data Path Width	64				
	LUTs	4050	4400	4050	4400	4400
	sysMEM EBRs	4				
	Registers	2800				
Design Tool Support	Lattice Implementation	Lattice Diamond™ 1.1 or ispLEVER® 8.1SP1				
	Synthesis	Synopsys® Synplify™ Pro for D-2010.03L-SP1				
	Simulation	Aldec® Active-HDL® 8.2 Lattice Edition				
		Mentor Graphics® ModelSim® SE 6.3F				

Features

- Compliant to IEEE 802.3-2005 standard, successfully passed University of New Hampshire InterOperability Laboratory (UNH-IOL) 10GbE MAC hardware tests[1]
- Supports standard 10Gbps Ethernet link layer data rate
- Supports rates up to 12Gbps by over-clocking
- 64-bit wide internal data path operating at 156.25MHz to 187.5MHz (187.5Mhz supported only on LatticeSC/SCM)
- XGMII interface to the PHY layer (using IODDR external to the core)
- XAUI interface to the PHY layer (using PCS/SERDES external to the core)
- Simple FIFO interface with user's application
- Optional Multicast address filtering

1. Successfully passed all UNH-IOL Clause 4 (MAC), Clause 31 (Flow Control) and Clause 46 (Reconciliation Sublayer) testing.

- Transmit and receive statistics vector
- Optional statistics counters of length from 16 to 40 bits for all devices (statistic counters are external to the core)
- Programmable Inter Frame Gap
- Supports:
  - Full duplex operation
  - Flow control using PAUSE frames
  - VLAN tagged frames
  - Automatic padding of short frames
  - Optional FCS generation during transmission
  - Optional FCS stripping during reception
  - Jumbo frames up to 16k
  - Inter frame Stretch Mode during transmission
  - Deficit Idle Count

Data rates up to 12Gbps are supported by increasing the 10 Gb+ Ethernet MAC system clock rate from the standard frequency of 156.25MHz used for processing 10Gbps data up to frequencies as high as 187.50MHz.

Functional Description

This chapter provides a functional description of the 10 Gb+ Ethernet MAC IP core. Figure 2-1 shows a top-level interface diagram for the 10 Gb+ Ethernet MAC IP.

Figure 2-1. 10 Gb+ Ethernet MAC Core Block Diagram

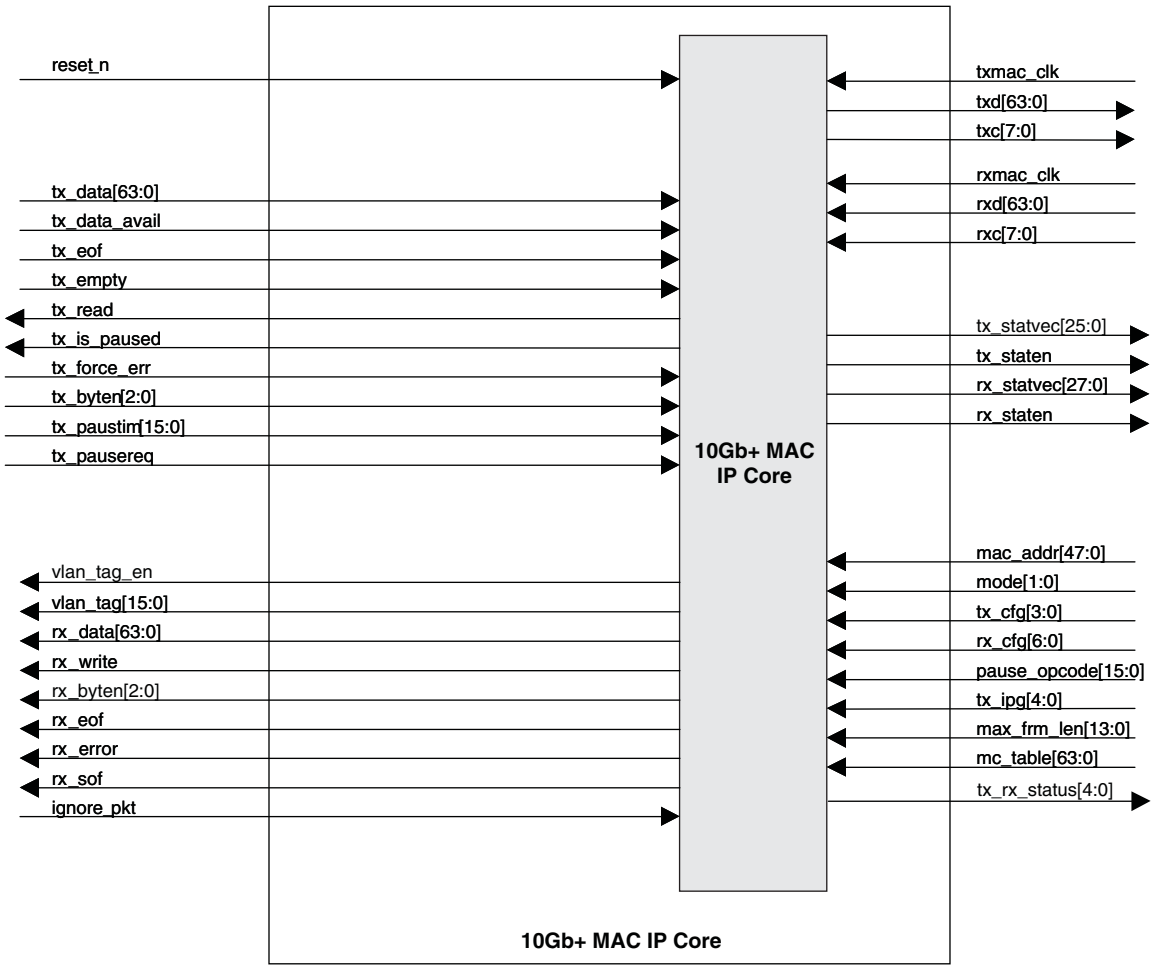
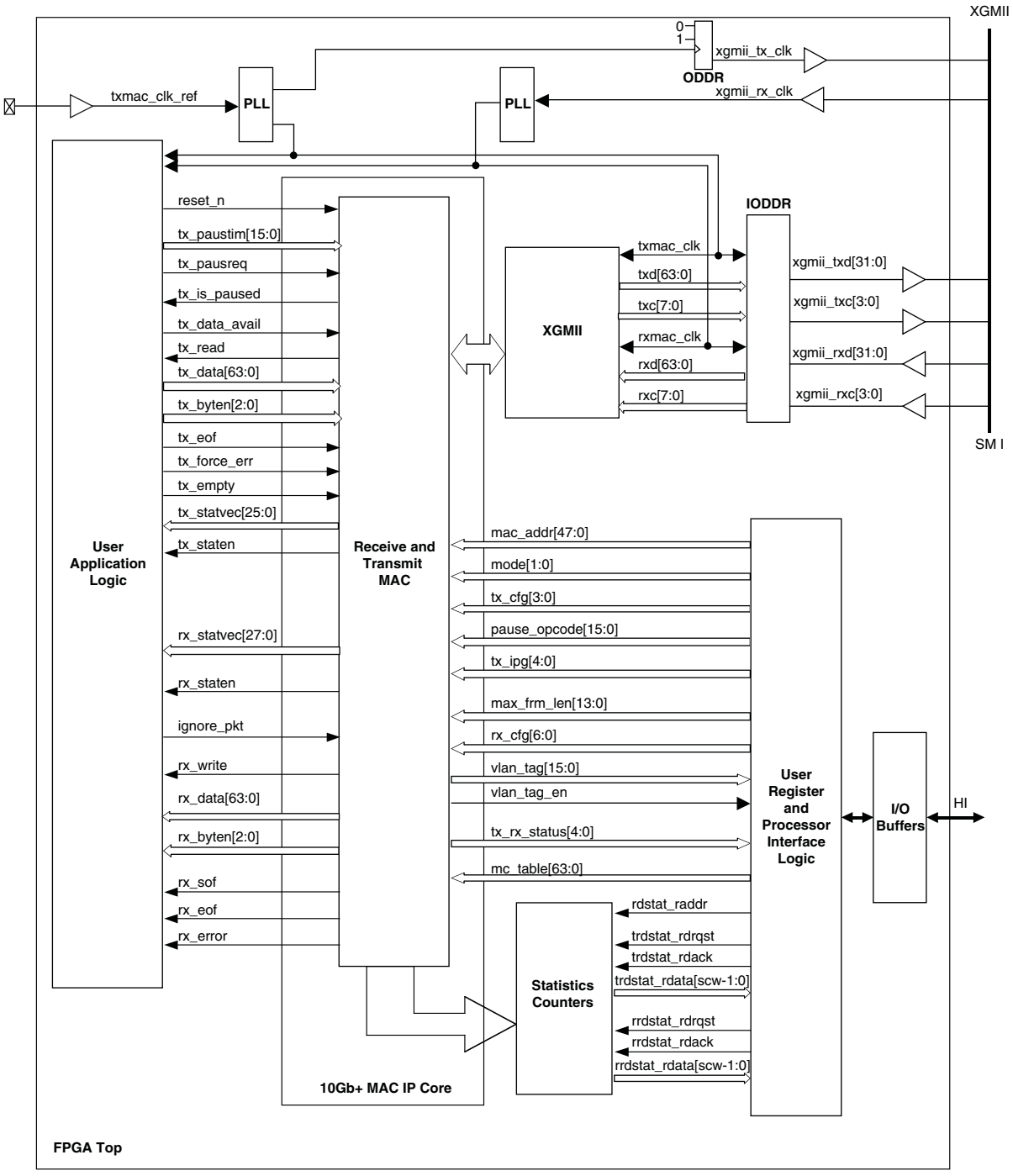


Figure 2-2 shows a system block diagram for the 10 Gb+ Ethernet MAC IP core.

Figure 2-2. 10 Gb+ Ethernet MAC Core System Block Diagram



The 10 Gb+ Ethernet MAC transmits and receives data between a host processor and an Ethernet network. The main function of the 10 Gb+ Ethernet MAC is to ensure that the Media Access rules specified in the 802.3 IEEE standard are met while transmitting a frame of data over Ethernet. On the receive side, the Ethernet MAC extracts the different components of a frame and transfers them to higher applications through a FIFO interface.

The format of an untagged Ethernet frame is shown in Figure 2-3. The format of a tagged Ethernet frame is shown in Figure 2-4.

Figure 2-3. Untagged Ethernet Frame Format

SOF 1 byte	PREAMBLE 6 bytes	SFD 1 byte	DESTINATION ADDRESS 6 bytes	SOURCE ADDRESS 6 bytes	LENGTH/ TYPE 2 bytes	DATA/ PAD 46-1500 bytes	FRAME CHECK SEQUENCE 4 bytes

Figure 2-4. Tagged Ethernet Frame Format

SOF 1 byte	PREAMBLE 6 bytes	SFD 1 byte	DESTINATION ADDRESS 6 bytes	SOURCE ADDRESS 6 bytes	TAG FIELD 4 bytes	LENGTH/ TYPE 2 bytes	DATA/ PAD 46-1500 bytes	FRAME CHECK SEQUENCE 4 bytes

The MAC is responsible for constructing a valid frame from the data received from the user's application before transmitting it. On the receive path, it receives frames from the network through the XGMII interface and passes the parameters of the frame to the MAC client (FIFO interface). The transmit path logic accepts the frame from the user application through a FIFO interface. The fields of the Ethernet frame that are expected from the application interface are shown in Figures 2-5 and 2-6.

On the transmit path, the MAC can be programmed in one of two modes. In the first mode (Figure 2-5), when the frame from the user contains the FCS along with the DESTINATION ADDRESS, SOURCE ADDRESS, LENGTH / TYPE and Data the Transmit MAC adds the SOF, preamble and SFD before transmitting the frame. This mode can be set by enabling the tx_pass_fcs bit in the TX_CTL register. In the second mode (Figure 2-6), the MAC calculates the number of bytes to be padded as well (if required) in addition to the FCS for the entire frame and adds the SOF, preamble and SFD before transmitting the frame.

Similarly, on the receive path, the MAC can be programmed to transfer the frame as it was received to the FIFO (Promiscuous mode) or after stripping off the FCS and any pad fields. In all cases the SOF, preamble and SFD bytes will always be stripped off the frame before it is transferred to the FIFO.

Figure 2-5. Ethernet Frame with Frame Check Sequence

DESTINATION ADDRESS	SOURCE ADDRESS	LENGTH/ TYPE	DATA/PAD 46-1500 bytes	FRAME CHECK SEQUENCE

Figure 2-6. Ethernet Frame with out Frame Check Sequence

DESTINATION ADDRESS	SOURCE ADDRESS	LENGTH/ TYPE	DATA/PAD 46-1500 bytes

The core has a number of inputs, which are used to provide control of the various modes of operation. Registers have been included at the top level of the FPGA. These registers are not part of the core. They are discussed in an appendix. The functionality of the MAC core is described in the following sections.

Receive MAC

The receive MAC is responsible for receiving the incoming frames and transferring them to the FIFO. In the process, it performs the following operations:

- Checks the frame for a valid SOF and SFD symbol.
- Determines whether the frame should be received by analyzing the Destination Address.
- Determines the type of the frame by analyzing the Length/Type field.
- Checks for any errors in the frame by recalculating the CRC and comparing it with the expected value.

The user can specify whether the FCS field should be written into the FIFO by programming the receive MAC configuration register. If the FCS field is to be stripped off the frame, any padding bytes within the frame will be stripped off as well.

Once a valid start of frame is detected, the destination address field of the incoming frame is analyzed. If the destination address was a unicast address, it is compared with the programmed MAC address. Unless the PRMS bit was set in the RX_CTL register, the incoming frame will be discarded if the destination address field and the programmed MAC address do not match.

If the frame had a multicast address and if receive_all_mc signal is not asserted, all such frames are dropped (except pause frames). If the frame had a multicast address and if receive_all_mc signal is asserted, the multicast frames are subject to address filtering rules as described below.

For all frames with multicast address, the CRC of the destination address is computed and the mid six bits of the least significant byte of the CRC is chosen as the address to a hash table. The MAC implements an eight row table with eight bits in each row. The lower three bits of the selected CRC are used to select one of these eight rows and the next three bits are used to select one of the bits in the selected table. The incoming multicast frame is accepted if the bit selected from the hash table was set to one. It is discarded if the bit selected was zero.

If the incoming frame had a broadcast address it will be accepted if the either the prms or the receive_bc bit was set. A broadcast frame is discarded if none of these signals are set.

A statistics vector that provides information about the incoming frame is generated when an incoming frame is received and transferred to the FIFO. A vector is not generated for all those frames that are discarded (No address match or frame length less than 64 bytes) or ignored (user asserts the ignore_pkt signal).

The vector is used to drive the management information block counters that keep track of all the happenings on the line. The composition of the statistics vector is shown in Table 2-1.

Table 2-1. Rx MAC Statistics Vector

Bit	Description
25	Receive Packet Ignored
24	Minimum IPG Violated
23	Unsupported Opcode
21	Frame Too Long
20	In Range Length Error
19	PAUSE Frame
18	VLAN Tag Detected
17	CRC Error
16	Multicast Frame Received
15	Broadcast Frame Received
13:0	Frame Byte Count

The full condition of the FIFO can be avoided by asserting the ignore_pkt signal. At the time a new frame is received, the receive logic samples this signal to determine whether the frame should be received.

Latency

Since the frame is buffered before being sent to the FIFO, there will be an initial latency. The first 64 bytes of a frame are buffered before they are sent out. Since the internal data path is 64 bits wide, the latency from the time a frame appears at the XGMII input to the time it begins to transfer to the FIFO will be eight clock cycles.

Transmit MAC

The main functions of the Transmit MAC are:

- Data padding for short frames when FCS generation is enabled.
- Generation of a pause frame when the tx_pausreq signal is asserted.
- To stop frame transmission when a Pause frame is received by the Receive MAC.
- Implement link fault signaling logic and transmit appropriate sequences based on the remote link status.

The TX_CTL register controls the operation of this module. For every frame transmitted, a statistics vector is generated. The composition of the vector is shown in Table 2-2.

Table 2-2. Transmit MAC Statistics Vector

Bit	Description
25	Long Frame Error
24	Terminate Error
23	Length Check Error
22	CRC Error
21	Underrun Error
20	Multicast Address
19	Broadcast Address
18	Tagged Frame
17	Jumbo Frame
16	MAC Control Inserted by Client
15	MAC Control Inserted by MAC
14	Transmit OK
13-0	Transmitted Bytes

Statistics Counters

This module provides counters for all of the bits in the statistics vectors. It also implements statistic counters from RFC2819 and RMON. The width of the counters is specified through a parameter and can have a value from 16 to 40 bits for all devices. The counters are read through an interface that returns the complete counter value in response to a read request. An acknowledgement is returned to indicate that the counter value on the read data bus is valid. There are separate request, acknowledge, and data buses for transmit counters and receive counters. The address bus is shared for transmit and receive counters. The Statistics Counter is a part of the reference design. A list of statistic counters is provided in Table 5-1 on page 25.

PHY Interface

The Reconciliation sublayer interface is implemented as a 64-bit wide single edge data bus and an 8-bit wide single edge control bus. To allow this interface to conform to the XGMII definition DDR I/O cells are added at the top level of the FPGA when this core is implemented. To allow this interface to conform to the XAUI definition a PCS/SERDES quad is added at the top level of the FPGA when this core is implemented. The top level file provided with this core contains the logic to implement one of these choices.

Register Description

There are no registers in this core. All control and status information is passed between this core and the top level of the device through individual I/O ports on the core. Registers must be added to the top level to control and monitor these ports. A reference description of a set of registers to do this is included as an appendix to this user's guide.

Signal Descriptions

Table 2-3. 10 Gb+ Ethernet MAC IP Core Input and Output Signals

Port Name	Active State	I/O Type	Description
reset_n	Low	Input	Asynchronous reset signal – Resets the entire core when asserted.
tx_paustim[15:0]	N/A	Input	Contains parameters to be transmitted in a pause frame. Valid when tx_pausreq is asserted.
tx_pausreq	Positive Edge	Input	Asserted to initiate the transmitting of a pause frame.
tx_is_paused	High	Output	Active when the transmitter has been placed in the pause state by a received pause frame.
tx_data_avail	High	Input	Asserted to alert the MAC transmitter that the transmit FIFO has data ready for transmission.
tx_read	High	Output	Transmit FIFO read request, asserted by the MAC transmitter in response to signal tx_data_avail.
tx_data[63:0]	N/A	Input	Data read from transmit FIFO. The least significant byte (bits 7:0) is sent out on the link first.
tx_byten[2:0]	N/A	Input	Indicates the valid bytes on the tx_data bus. Must be 7 except when tx_eof is active. 0 - tx_data[7:0] valid 1 - tx_data[15:0] valid 2 - tx_data[23:0] valid 3 - tx_data[31:0] valid 4 - tx_data[39:0] valid 5 - tx_data[47:0] valid 6 - tx_data[55:0] valid 7 - tx_data[63:0] valid
tx_eof	High	Input	End of frame signal asserted with the last segment of the frame.
tx_force_err	High	Input	Indicates that the current frame has errors. This input is qualified with input signal tx_eof.
tx_empty	High	Input	When asserted, indicates that the transmit FIFO is empty.
tx_statvec[25:0]	N/A	Output	Contains information on the frame transmitted (details given in the Functional Description section of this document). This bus is qualified by the tx_staten signal.
tx_staten	High	Output	When asserted, indicates that the contents of the tx_statvec bus are valid. This signal is asserted for 3 txmac_clk periods.
rx_statvec[27:0]	N/A	Output	Contains information on the frame received (details given in the Functional Description section of this document). This bus is qualified by the rx_staten signal.
rx_staten	High	Output	When asserted, indicates that the contents of the rx_statvec bus are valid. This signal is asserted for 3 rxmac_clk periods.
ignore_pkt	High	Input	Asserted to prevent a receive FIFO full condition. The Receive MAC will continue dropping packets as long as this signal is asserted.
rx_write	High	Output	Driven by the MAC core to request a receive FIFO write.
rx_data[63:0]	N/A	Output	Contains data that is to be written into the receive FIFO.
rx_byten[2:0]	N/A	Output	Indicates the valid bytes on the rx_data bus. Must be 7 except when rx_eof is active. 0 - rx_data[7:0] valid 1 - rx_data[15:0] valid 2 - rx_data[23:0] valid 3 - rx_data[31:0] valid 4 - rx_data[39:0] valid 5 - rx_data[47:0] valid 6 - rx_data[55:0] valid 7 - rx_data[63:0] valid
rx_sof	High	Output	Start of frame signal asserted with the first segment of the frame.

Table 2-3. 10 Gb+ Ethernet MAC IP Core Input and Output Signals (Continued)

Port Name	Active State	I/O Type	Description
rx_eof	High	Output	End of frame signal asserted with the last segment of the frame.
rx_error	High	Output	When asserted, indicates that the frame had a length error, a termination error, or a CRC error. This signal is qualified with the rx_eof signal.
txmac_clk	N/A	Input	156.25 MHz MAC transmit clock. Ethernet frame transmit data is output on the rising edge of this clock.
txd[63:0]	N/A	Output	Single data rate Ethernet frame transmit data. Converted to double data rate XGMII transmit data signals with FPGA ODDR circuit elements external to the IP core.
txc[7:0]	High=control Low=data	Output	Asserted by the MAC to indicate on a per byte basis that the txd bus contains data or control. Converted to double data rate XGMII transmit control signals with FPGA ODDR circuit elements external to the IP core.
rxmac_clk	N/A	Input	156.25 MHz MAC receive clock. Receive data is presented to the receive MAC coincident with the rising edge of this clock.
rxd[63:0]	N/A	Input	Single data rate Ethernet frame receive data. Converted from double data rate XGMII receive data signals with FPGA IDDR circuit elements external to the IP core.
rxc[7:0]	High=control Low=data	Input	Indicates on a per byte basis that the rxd bus contains data or control.
mac_addr[47:0]	N/A	Input	Unicast address for the MAC. This address is received from or sent to the link from most significant byte to least significant byte. For MAC address: AC-DE-48-00-00-80, AC is sent first and 80 is sent last.
mode[1:0]	High	Input	Bit 0 (rx_en) - Enables the receive side of the MAC. Bit 1 (tx_en) - Enables the transmit side of the MAC.
tx_cfg[3:0]	High	Input	Bit 0 (tx_pass_fcs) - When zero, the MAC generates and inserts FCS in outgoing packets. Bit 1 (transmit_pause_en) - Enables the transmit side of the MAC to support flow control. Bit 2 (tx_ipg_stretch) - Enables the transmit side of the MAC to insert the proper amount of inter-frame gap to support matching rate of OC192. Bit 3 (transmit_short) - Enables the transmit side of the MAC to transmit short frames.
pause_opcode[15:0]	N/A	Input	Supplies opcode for transmit pause frames.
tx_ipg[4:0]	N/A	Input	Specifies the amount of inter-frame gap in increments of 4 bytes. 0 - indicates the minimum value of 8 bytes.
max_frm_len[13:0]	N/A	Input	Specifies the maximum frame length. A frame longer than this will be marked as long.
rx_cfg[6:0]	High	Input	Bit 0 (prms) - Enables the receive side of the MAC to receive frames without doing any address filtering. Bit 1 (rx_pass_fcs) - When zero, the MAC discards the FCS of incoming packets after checking it. Bit 2 (rx_pause_en) - Enables the receive side of the MAC to support flow control. Bit 3 (receive_all_mc) - Enables the receive side of the MAC to receive multicast frames as per address filtering rules. Bit 4 (receive_bc) - Enables the receive side of the MAC to receive broadcast frames. Bit 5 (receive_short) - Enables the receive side of the MAC to receive short frames. Bit 6 (drop_mac_ctrl) - Enables the receive side of the MAC to drop MAC Control packets before passing them on to client.

Table 2-3. 10 Gb+ Ethernet MAC IP Core Input and Output Signals (Continued)

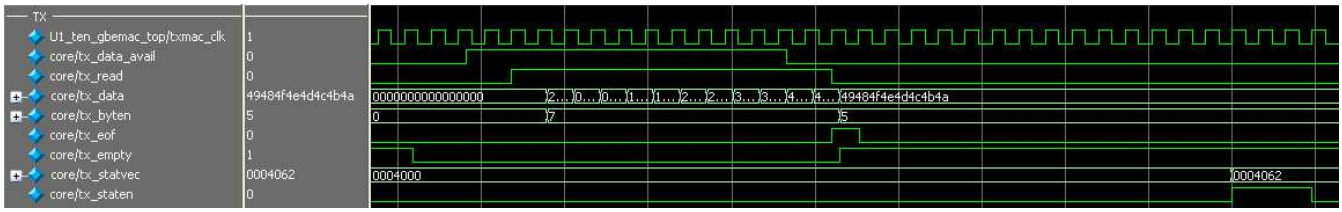
Port Name	Active State	I/O Type	Description
vlan_tag[15:0]	N/A	Output	The most recently received VLAN tag.
vlan_tag_en	High	Output	When asserted, indicates that the contents of the vlan_tag bus are valid. This signal is asserted for 3 rxmac_clk periods.
tx_rx_status[4:0]	High	Output	Bit 0 (tx_idle) - When asserted indicates that the transmitter is idle. Bit 1 (rx_idle) - When asserted indicates that the receiver is idle. Bits[4:2] (link_sts[2:0]) - Contains the RS layer status of the link. (1=local fault, 2= remote fault, 4=link ok)
Multicast Address Filtering Optional Signals			
mc_table[63:0]	N/A	Input	Multicast table

Timing Specifications

Transmit Interface

When there is a packet ready for transmission, the tx_data_avail is asserted. This signal should stay active until the MAC asserts the tx_read, after which MAC will ignore the status of tx_data_avail and so the tx_data_avail signal can be deactivated. The data is available one clock cycle after the MAC asserts the tx_read. The MAC continues reading the packet until it gets tx_eof active. The tx_eof signal is asserted when the last byte of data is transmitted. Internally the MAC buffers the data before it starts transmission of a packet. The MAC may stop/continue reading to maintain the level of the buffer due to factors like IPG insertion and Pause reception. Hence, the tx_read signal can be deactivated anytime during the transfer. Once the MAC receives tx_eof, the MAC will again start monitoring the tx_data_avail signal.

Figure 2-7. Transmission of a 64 Data Byte Frame

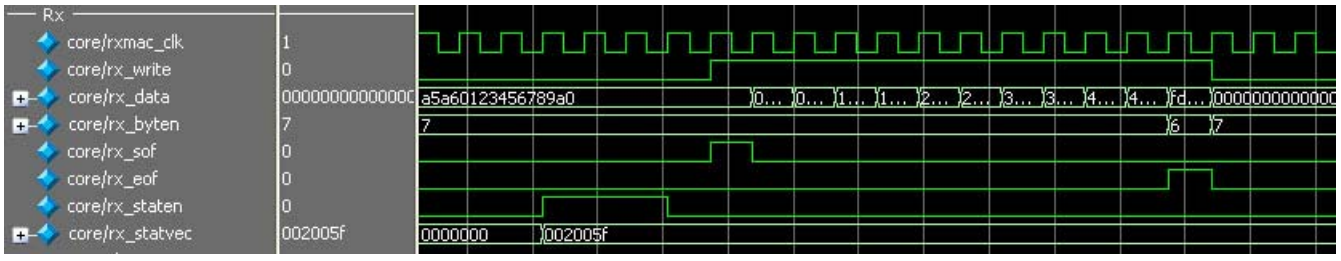


The tx_staten signal will be asserted once the entire frame is received from the user's interface.

Receive Interface

When the MAC receives the data, it asserts rx_write. The MAC asserts rx_sof when it writes the first byte of data. The rx_write signal may be deasserted anytime between a packet. In this case the value on rx_data signal is not valid. The end of a packet is indicated with the assertion of rx_eof signal.

Figure 2-8. Reception of a 64 Data Byte Frame



The IPexpress™ tool is used to create IP and architectural modules in the Diamond and ispLEVER software. Refer to "IP Core Generation" on page 16for a description on how to generate the IP.

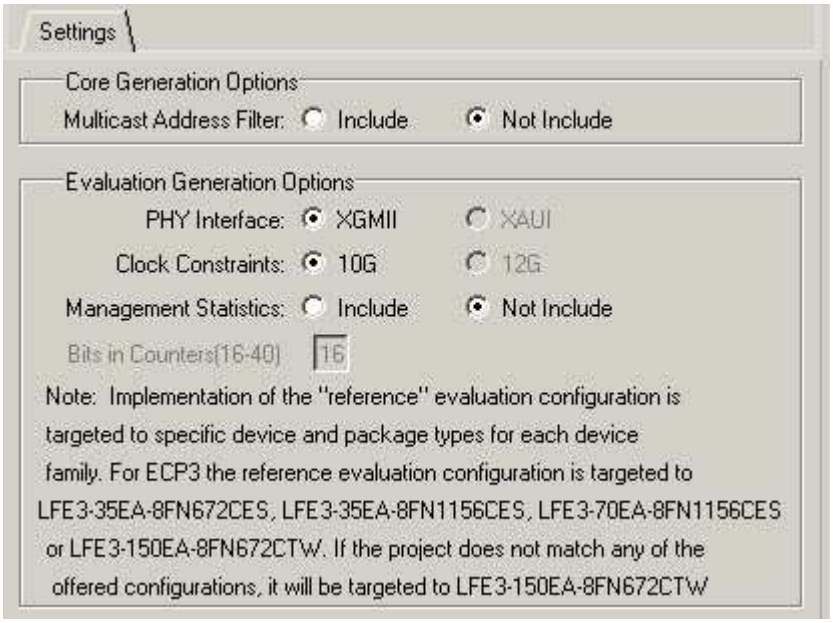
Table 3-1 provides the list of user configurable parameters for the 10 Gb+ Ethernet MAC IP core. The parameter settings are specified using the 10 Gb+ Ethernet MAC IP core Configuration GUI in IPexpress.

Table 3-1. 10 Gb+ Ethernet MAC IP Core Configuration Parameters

Parameter	Value Range	Default
Core Generation Options		
Multicast Address Filter	Include/Not Include	Not Include
Evaluation Generation Options		
PHY Interface	XGMI/XAUI	XGMI
Clock Constraints	10G/12G	10G
Management Statistics	Include/Not Include	Not Include
Bits in Counters[13-40]	16-40	16

Figure 3-1 shows the 10 Gb+ Ethernet MAC IP Core configuration dialog box.

Figure 3-1. 10 Gb+ Ethernet MAC IP Core Configuration Dialog Box



Parameter Descriptions

This section describes the available parameters for the 10 Gb+ Ethernet MAC IP core.

Multicast Address Filter

This parameter determines whether the optional Multicast Address Filtering will be included in the core's implementation.

Evaluation Generation Options

PHY Interface

The reconciliation sublayer interface is implemented as a 64 bit wide single edge data bus and an 8-bit wide single edge control bus. To allow this interface to conform to the XGMII definition DDR I/O cells are added at the top level of the FPGA when this core is implemented. To allow this interface to conform to the XAUI definition, a PCS/SERDES quad is added at the top level of the FPGA when this core is implemented. The top level file provided with this core contains the logic to implement one of these choices.

Clock Constraints

This parameter is used to specify whether timing constraints supporting either 10 Gbps or 12 Gbps data throughput are included in the evaluation package provided with the IP core.

Management Statistics

This parameter determines whether the optional Statistics Counters will be included in the reference design.

Bits in Counters[13-40]

This parameter determines the width of the optional Statistics Counters.

IP Core Generation

This chapter provides information on how to generate the 10 Gb+ Ethernet MAC IP core using the Diamond or ispLEVER software IPexpress tool, and how to include the core in a top-level design.

Licensing the IP Core

An IP core- and device-specific license is required to enable full, unrestricted use of the 10 Gb+ Ethernet MAC IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/products/intellectualproperty/aboutip/isplevercoreonlinepurchas.cfm

Users may download and generate the 10 Gb+ Ethernet MAC IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The 10 Gb+ Ethernet MAC IP core also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See "Hardware Evaluation" on page 23 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond or ispLEVER EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

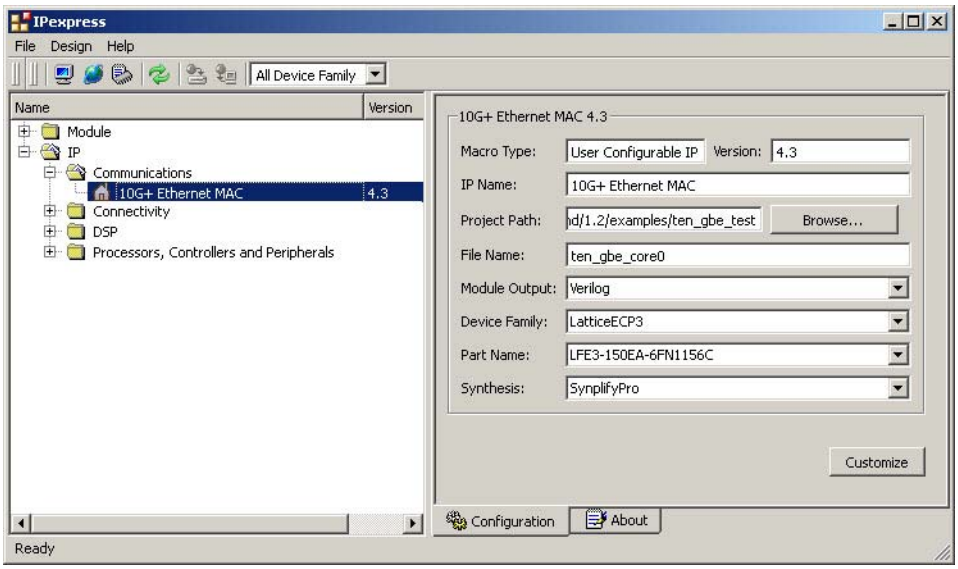
Getting Started

The 10 Gb+ Ethernet MAC IP core is available for download from the Lattice IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the 10 Gb+ Ethernet MAC IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL.
- **Device Family** – Device family to which IP is to be targeted (e.g. LatticeSCM, Lattice ECP2M, LatticeECP3, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

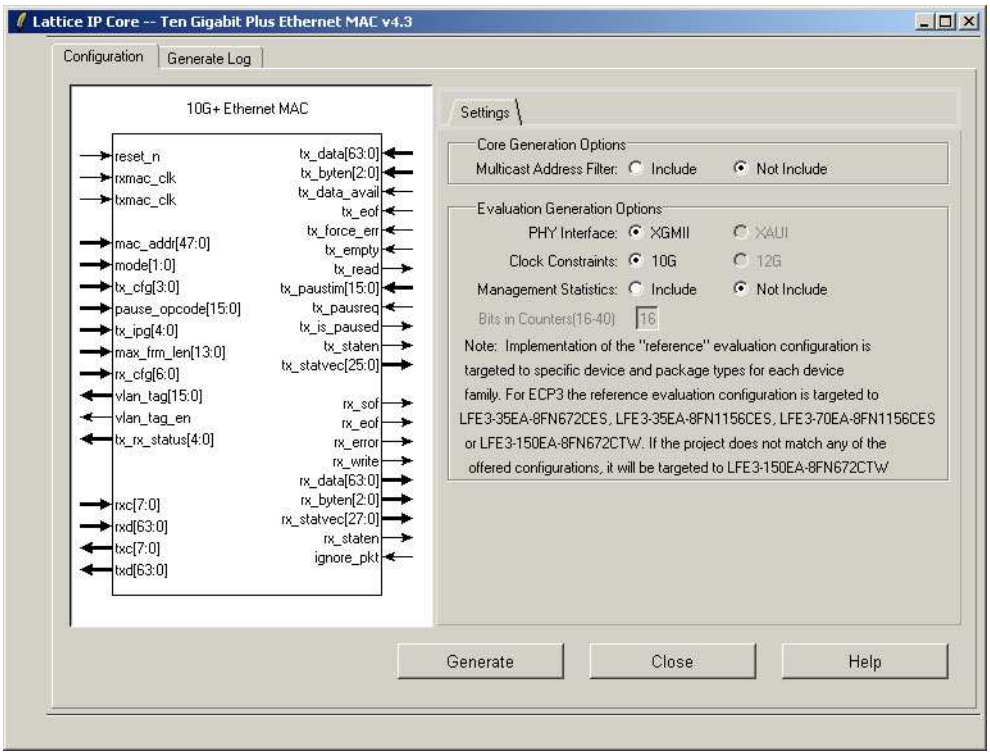
Figure 4-1. IPexpress Dialog Box (Diamond Version)



Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output (Design Entry in ispLEVER), Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the 10 Gb+ Ethernet MAC IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application. Refer to "Parameter Settings" on page 14 for more information on the 10 Gb+ Ethernet MAC IP core parameter settings.

Figure 4-2. Configuration GUI (Diamond Version)



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified “Project Path” directory. The directory structure of the generated files is shown in Figure 4-3.

Figure 4-3. LatticeECP3 10 Gb+ Ethernet MAC IP Core Directory Structure

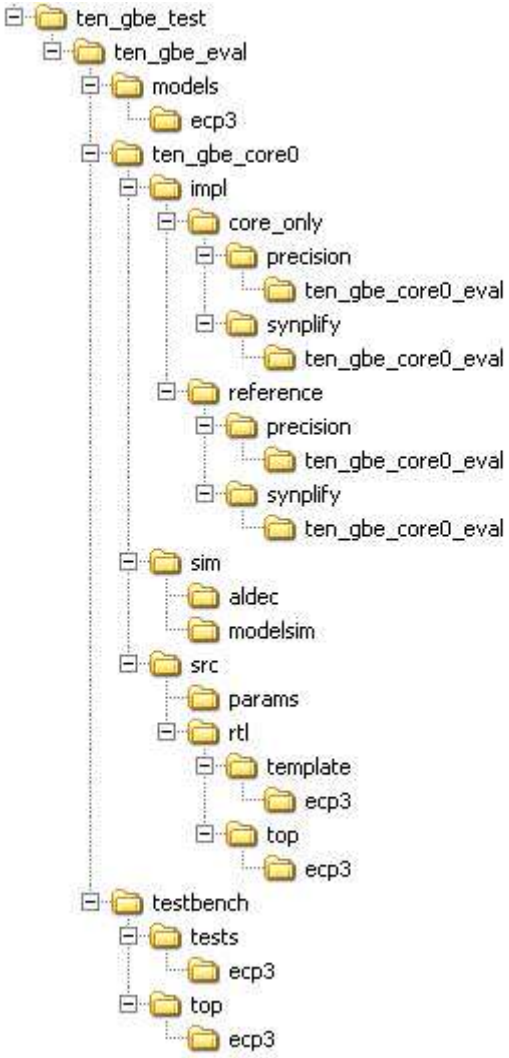


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<*username*>_inst.v	This file provides an instance template for the IP.
<*username*>.v	This file provides the 10 Gb+ Ethernet MAC core for simulation.
<*username*>_beh.v	This file provides a behavioral simulation model for the 10 Gb+ Ethernet MAC core.
<*username*>_params.v	This file provides parameters necessary for the simulation.
<*username*>_bb.v	This file provides the synthesis black box for the user's synthesis.
<*username*>.ngo	This file provides the synthesized IP core.
<*username*>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.

Table 4-1. File List (Continued)

File	Description
<*username*>.ipx	The IPX file holds references to all of the elements of an IP or Module after it is generated from the IPexpress tool (Diamond version only). The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP/Module generation GUI when an IP/Module is being re-generated.
<*username*>_top.[v,vhd]	This file provides a module which instantiates the 10 Gb+ Ethernet MAC core. This file can be easily modified for the user's instance of the 10 Gb+ Ethernet MAC core. This file is located in the <*username*>_eval/<*username*>_/src/rtl/top/ directory.

These are all of the files necessary to implement and verify the 10 Gb+ Ethernet MAC IP core in your own top-level design. The following additional files providing IP core generation status information are also generated in the "Project Path" directory:

- <*username*>_generate.log – Synthesis and map log file.
- <*username*>_gen.log – IPexpress IP generation log file.

The \<ten_gbe_eval> and subtending directories provide files supporting 10 Gb+ Ethernet MAC core evaluation. The \<ten_gbe_eval> directory shown in Figure 4-3 contains files/folders with content that is constant for all configurations of the 10 Gb+ Ethernet MAC. The \<*username*> subfolder (\ten_gbe_core0 in this example) contains files/folders with content specific to the username configuration.

The \ten_gbe_eval directory is created by IPexpress the first time the core is generated and updated each time the core is regenerated. A \<*username*> directory is created by IPexpress each time the core is generated and regenerated each time the core with the same file name is regenerated. A separate \<*username*> directory is generated for cores with different names, e.g. \<ten_gbe_core0>, \<ten_gbe_core1>, etc.

Instantiating the Core

The generated 10 Gb+ Ethernet MAC IP core package includes black-box (<*username*>_bb.v) and instance (<*username*>_inst.v) templates that can be used to instantiate the core in a top-level design. Two example RTL top-level reference source files are provided in \<*project_dir*>\ten_gbe_eval\<*username*>\src\rtl\top\sc.

The top-level file ten_gbemac_top.v is the same top-level that is used in the simulation model described in the next section. Designers may use this top-level reference as the starting template for the top-level for their complete design. Included in ten_gbemac_top.v is logic, memory and clock modules supporting an XGMII interface loop back capability, a register module supporting programmable control of the 10 Gb+ Ethernet MAC core parameters and system processor interface via the LatticeSC integrated SYSBUS capability. Verilog source RTL for these modules is provided in \<project_dir>\ten_gbe_eval\<username>\src\rtl\template\sc. The top-level configuration is specified via the parameters defined in the ten_gbemac_defines.v file in \<*project_dir*>\ten_gbe_eval\<*username*>\src\params. A description of the 10 Gb+ Ethernet MAC parameters is in the parameters section of this document. A description of the 10 Gb+ Ethernet MAC register layout for this reference design is provided in an appendix to this document.

The top-level file ten_gbemac_core_only_top.v supports the ability to implement just the 10 Gb+ Ethernet MAC core itself. This design is intended only to provide an accurate indication of the device utilization associated with the 10 Gb+ Ethernet MAC core and should not be used as an actual implementation example.

Running Functional Simulation

The functional simulation includes a configuration-specific behavioral model of the 10 Gb+ Ethernet MAC IP Core (<username>_beh.v) that is instantiated in an FPGA top level along with a client-side interface loop back capability module and register implementation module.

The top-level file supporting ModelSim eval simulation is provided in \<*project_dir*>\ten_gbe_eval\<*username*>\sim\modelsim. This FPGA top is instantiated in an eval testbench provided in \<*project_dir*>\ten_gbe_eval\testbench\sc that configures FPGA test logic registers and 10 Gb+ Ethernet MAC IP core control and status registers via an included test file stimulus_file.v provided in \<*project_dir*>\ten_gbe_eval\testbench\tests\sc. Note the user can edit the stimulus_file.v file to configure and monitor whatever registers they desire.

Users may run the eval simulation by doing the following:

1. Open ModelSim.

2. Under the File tab, select **Change Directory** and choose folder \<*project_dir*>\ten_gbe_eval\<*username*>\sim\modelsim.

3. Under the Tools tab, select **Execute Macro** and execute one of the ModelSim “do” scripts shown.

The top-level file supporting Aldec Active-HD® simulation is provided in \<*project_dir*>\ten_gbe_eval\<*username*>\sim\aldec. This FPGA top is instantiated in an eval testbench provided in \<*project_dir*>\ten_gbe_eval\testbench\sc that configures FPGA test logic registers and 10 Gb+ Ethernet MAC IP core control and status registers via an included test file stimulus_file.v provided in \<*project_dir*>\ten_gbe_eval\testbench\tests\sc. Note the user can edit the stimulus_file.v file to configure and monitor whatever registers they desire.

Users may run the eval simulation by doing the following:

1. Open Active-HDL.

2. Under the Console tab, change the directory to: <*project_dir*>\ten_gbe_eval\<*username*>\sim\aldec??

3. Execute the Active-HDL “do” scripts shown.

The simulation waveform results will be displayed in the Aldec Active-HDL Wave window.

Synthesizing and Implementing the Core in a Top-Level Design

The 10 Gb+ Ethernet MAC IP core itself is synthesized and is provided in NGO format when the core is generated. Users may synthesize the core in their own top-level design by instantiating the core in their top level as described previously and then synthesizing the entire design with either Synplify or Precision RTL Synthesis.

Two example RTL top-level configurations supporting 10 Gb+ Ethernet MAC core top-level synthesis and implementation are provided with the 10 Gb+ Ethernet MAC IP core in \<*project_dir*>\ten_gbemac_eval\<*username*>\impl.

The top-level file ten_gbemac_core_only_top.v provided in \<*project_dir*>\ten_gbemac_eval\<*username*>\src\rtl\top supports the ability to implement just the 10 Gb+ Ethernet MAC core. This design is intended only to provide an accurate indication of the device utilization associated with the core itself and should not be used as an actual implementation example.

The top-level file ten_gbemac_reference_top.v provided in \<*project_dir*>\ten_gbemac_eval\<*username*>\src\rtl\top supports the ability to instantiate, simulate, map, place and route the Lattice 10 Gb+ Ethernet MAC IP core in a complete example design. This reference design basically provides a loopback path for packets on the MAC Rx/Tx client interface, through a FIFO and associated logic. Ethernet packets are sourced to the Rx XGMII and looped back on the MAC Rx/Tx Client FIFO interface. Source and destination addresses in the Ethernet frame can be swapped so the looped back packets on the Tx XGMII have the correct source and destination addresses. This is the same configuration that is used in the evaluation simulation capability described previously. Note that implementation of the reference evaluation configuration is targeted to a specific device and package type for each device family.

Specifically:

- LatticeECP2: LFE235E-7F672C
- LatticeECP2S: LFE235ES-7F672C
- LatticeECP2M: LFE2M35E-7F672C
- LatticeECP2MS: LFE2M35SE-7F672C
- LatticeECP3: LFECP370E-8F1156CES
- LatticeECP3: LFECP3-35E-8F1156CES
- LatticeECP3: LFE3-150EA –8F 672CTW
- LatticeSC: LFSC3GA25E-5F900C
- LatticeSCM: LFSCM3GA25EP1-5F900C

Push-button implementation of both top-level configurations is supported via the Diamond or ispLEVER project files, <username>_reference_eval.syn and <username>_core_only_eval.syn. These files are located in \<*project_dir*>\ten_gbemac_test\ten_gbemac_eval\<*username*>\impl\<configuration>.

For the Linux platform, the top-level synthesis must be run separately with a synthesis tool such as Synplify Pro, since Diamond or ispLEVER for Linux only accepts an EDIF entry. For the core only project, synthesis tcl files will be generated for this purpose, including <username>_core_only_top.tcl in the directory \<project_dir\ten_gbemac_eval\<*username*>\impl\core_only\synplify. The user can run this tcl script to synthesize the core_only top_level files in the above directory.

For the reference project, <username>_reference_top.tcl will be generated in the \<*project_dir*>\ten_gbemac_eval\<*username*>\impl\reference\synplify directory. The user can run this tcl script to synthesize the reference top_level files in the above directory.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to \<*project_dir*>\ten_gbemac_eval\<*username*>\impl\synplify (or precision) in the Open Project dialog box.
3. Select and open <*username*>.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

To use this project file in ispLEVER:

1. Choose **File > Open Project.**
2. Browse to \<*project_dir*>\ten_gbemac_eval\<*username*>\impl\synplify (or precision) in the Open Project dialog box.
3. Select and open <*username*>.syn. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the device top-level entry in the left-hand GUI window.

5. Implement the complete design via the standard ispLEVER GUI flow.

Hardware Evaluation

The 10 Gb+ Ethernet MAC IP core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Enabling Hardware Evaluation in ispLEVER

In the Processes for Current Source pane, right-click the **Build Database** process and choose **Properties** from the dropdown menu. The hardware evaluation capability may be enabled/disabled in the Properties dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate.** The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.
9. Check the Generate Log tab to check for warnings and error messages.
10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is

already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Regenerating an IP Core in ispLEVER

To regenerate an IP core in ispLEVER:

1. In the IPexpress tool, choose **Tools > Regenerate IP/Module**.
2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core you wish to regenerate, and click **Open**.
3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.
4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.
5. Click **Next**. The IP core's dialog box opens showing the current option settings.
6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.
7. Click **Generate**.
8. Click the **Generate Log** tab to check for warnings and error messages.

This chapter gives application support information for the 10 Gb+ Ethernet MAC IP core.

Reference Register Descriptions

There are no registers in this IP core. All control and status information is passed between the core and the top level of the device through individual I/O ports on the core. Registers must be added to the top level to control and monitor these ports. This appendix describes all the registers needed to control and monitor the 10 Gb+ MAC IP core and the test logic at the top level. Unused bits of all the registers cannot be written. When read, the unused bits return a value of zero. The default values of the registers are restored when the core is reset.

All the registers except the MODE register should be written only after disabling the MAC. All of the registers can be read at any time.

Table 5-1. 10 Gb+ Ethernet MAC IP Core Internal Registers

Internal Registers			
Register Description	**Mnemonic**	**I/O Address**	**Reset Value**
Version Register	VERID	A00H	X1H
Mode Register	MODE	A01H	00H
Transmit Control Register	TX_CTL	A02H	00H
Receive Control Register	RX_CTL	A03H	00H
Maximum Packet Size Register	MAX_PKT_SIZE_0	A04H	05H
Maximum Packet Size Register	MAX_PKT_SIZE_1	A05H	EEH
Inter Packet Gap	IPG_VAL	A06H	01H
MAC Address 0	MAC_ADDR_0	A07H	00H
MAC Address 1	MAC_ADDR_1	A08H	00H
MAC Address 2	MAC_ADDR_2	A09H	00H
MAC Address 3	MAC_ADDR_3	A0AH	00H
MAC Address 4	MAC_ADDR_4	A0BH	00H
MAC Address 5	MAC_ADDR_5	A0CH	00H
Transmit Receive Status Register	TX_RX_STS	A0DH	00H
VLAN tag Length/Type Register	VLAN_TAG_0	A0EH	00H
VLAN tag Length/Type Register	VLAN_TAG_1	A0FH	00H
Multicast_table_0	MC_TAB_0	A10H	00H
Multicast_table_1	MC_TAB_1	A11H	00H
Multicast_table_2	MC_TAB_2	A12H	00H
Multicast_table_3	MC_TAB_3	A13H	00H
Multicast_table_4	MC_TAB_4	A14H	00H
Multicast_table_5	MC_TAB_5	A15H	00H
Multicast_table_6	MC_TAB_6	A16H	00H
Multicast_table_7	MC_TAB_7	A17H	00H
Pause Opcode	PAUS_OP_0	A18H	00H
Pause Opcode	PAUS_OP_1	A19H	01H
Test Control Register	TSTCNTL	B00H	02H
MAC Control Register	MACCNTL	B01H	00H
Pause Time Register	PAUSTMR_0	B02H	04H
Pause Time Register	PAUSTMR_1	B03H	FFH