Design Gateway Co.,Ltd
54 BB Building 14th Fl., Room No.1402 Sukhumvit 21 Rd. (Asoke), Klongtoey-Nua, Wattana, Bangkok 10110
Phone: 66(0)2-261-2277
Fax: 66(0)2-261-2290
E-mail: ip-sales@design-gateway.com
URL: www.design-gateway.com

Core Facts

Provided with Core	
Documentation	User Guide, Design Guide
Design File Formats	Encrypted HDL
Constraints Files	User constraint file
Verification	Test Bench, Simulation Library
Instantiation Templates	VHDL
Reference Designs & Application Notes	Vivado Project, See Reference Design Manual
Additional Items	Demo on ZCU102, ZCU106
Simulation Tool Used	
ModelSim SE	
Support	
Support Provided by Design Gateway Co., Ltd.	

Features

- TCP/IP stack implementation
- Support IPv4 protocol
- Support one session by one TOE40G-IP (Multisession could be implemented by using multiple TOE40G-IPs)
- Support both Server and Client mode
- Support Jumbo frame
- Simple user interface for control signal by standard register interface
- Simple user interface for data signal by 256-bit standard FIFO interface
- Transmit packet size must be aligend to 32-byte to match Tx FIFO interface
- Total receive data must be aligend to 32-byte to match Rx FIFO interface
- Adjustable transmit/receive buffer size to optimize resource and performance
- 256-bit FIFO interface with Ethernet MAC
- At least 200 MHz is recommended for clock input to TOE40G-IP
- Reference design available on ZCU102 and ZCU106 evaluation board
- Not support data fragmentation feature

Table 1: Example Implementation Statistics for Ultrascale device

Family	Example Device	Fmax (MHz)	CLB Regs	CLB LUTs	CLB[1]	IOB	BRAMTile[2]	Design Tools
Zynq-Ultrascale+	XCZU9EG-FFVB1156-2	300	5268	5271	996	-	34.5	Vivado2017.4

Notes:

1) Actual logic resource dependent on percentage of unrelated logic

2) Block memory resources are based on 64kB Tx data buffer size, 16kB Tx packet buffer size, and 64kB Rx data buffer size. Minimum size of each buffer is 16kB.

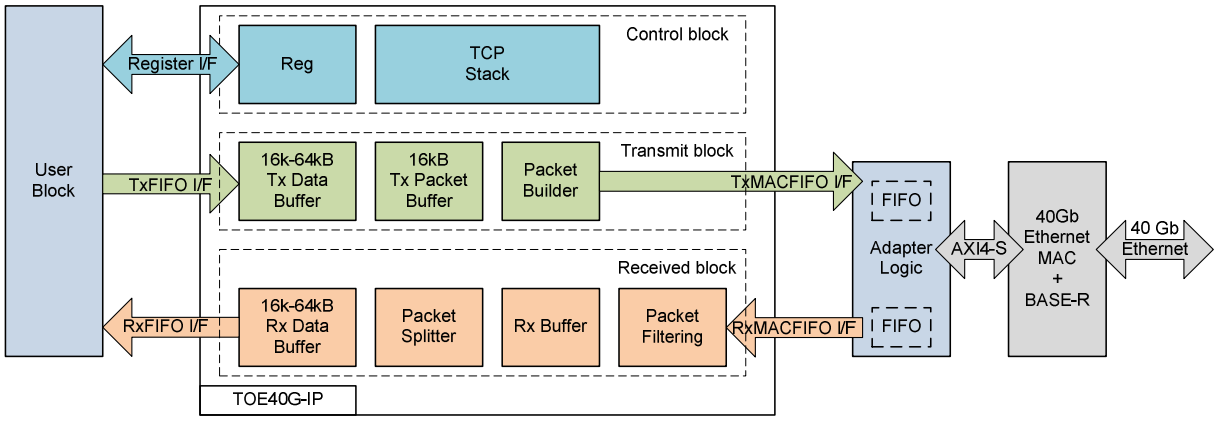

Figure 1: TOE40G-IP Block Diagram

Applications

TOE40G-IP is designed for network application using TCP/IP protocol for data reliability with ultra high speed performance by using 40 Gb Ethernet. Ethernet logger or network equipment could be designed by using TOE40G-IP without CPU requirement.

General Description

TOE40G-IP core implements TCP/IP stack. It needs to connect with 40 Gb Ethernet MAC and PCS/PMA (BASE-R) module from Xilinx which implements lower layer network protocol. User logic sends and receives 40 Gb Ethernet data with network equipments by designing simple logic to interface with TOE40G-IP.

User interface could be split into two groups, i.e. control signals and data signals. Control signals are designed by using general register interface while data signals are designed by using 256-bit general FIFO interface.

Register interface is used to set network parameters such as MAC address, port number, IP address. Also, it uses to send command to the IP such as open connection (active mode), send data, and close connection (active mode). The parameters for send data command such as transfer size and packet size are also set through Register interface.

Data interface with user logic uses 256-bit FIFO interface for both Tx and Rx direction. User could design simple logic to send data to TOE40G-IP or receive data from TOE40G-IP following general FIFO timing diagram.

Following TCP/IP protocol, the new connection must be opened before transferring data between two devices. TOE40G-IP supports to open connection as active mode (port is opened by TOE40G-IP) and passive mode (port is opened by external device). After new connection is established, TOE40G-IP can send or receive data with external device through the new port number. When all data have been transferred completely, the connection could be closed by TOE40G-IP (active mode) or by external device (passive mode).

Data interface

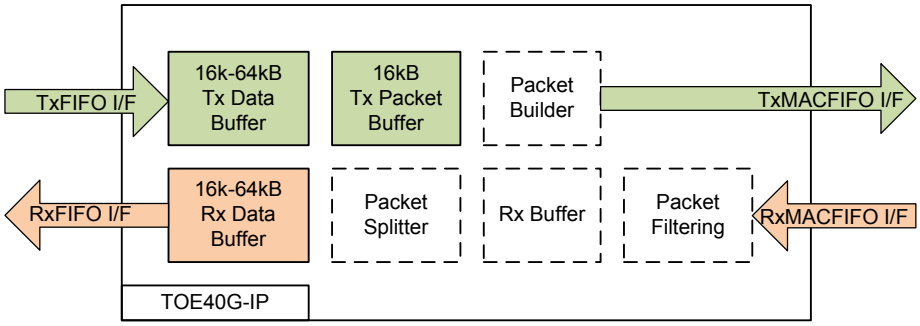

Figure 2: Adjustable Tx/Rx Buffer Size

To accelerate transfer speed, three buffers are applied in TOE40G-IP, i.e. Tx data buffer, Tx packet buffer, and Rx data buffer as shown in Figure 2. User selects the size of Tx data buffer and Rx data buffer as input parameter of TOE40G-IP HDL code. Using bigger data buffer size could increase transfer performance. It is a trade-off between transfer performance and FPGA resource utilization.

Tx data buffer size is effect to Tx performance of TOE40G-IP (TOE40G-IP to external device) while Rx data buffer size is effect to Rx performance of TOE40G-IP (external device to TOE40G-IP). Tx packet buffer in TOE40G-IP is fixed to 16 Kbyte size.

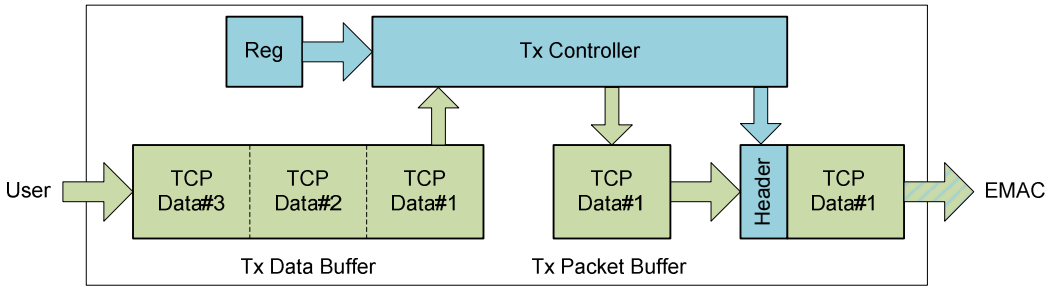

Figure 3: Transmit Data Flow

To include data in TCP packet, one packet data from Tx data buffer is forwarded to Tx packet buffer. Network parameters are loaded from Reg to calculate the header of each packet within Tx controller. After header processing is completed, the header is inserted at the beginning of packet and following by one packet data. Complete TCP packet is forwarded to EMAC. TCP checksum and IP checksum within the header are also calculated in Tx controller. Busy flag of TOE40G-IP (read through Register I/F) is de-asserted to '0' after complete data transmission to external device. Total data and packet size could be set by user through Register I/F.

During sending data from TOE40G-IP to external device, the received packet from EMAC is monitored to check receive buffer status of external device. TOE40G-IP could send data when received buffer size within external device has free space enough (more than packet size). Acknowledge number of received packet is also monitored to monitor the latest data position which has been received completely. Following TCP/IP standard, when duplicate ACK is received (acknowledge number is same as previous packet), TOE40G-IP will retransmit the lost packet to external device. The lost position is referred by the acknowledge number within duplicate ACK packet.

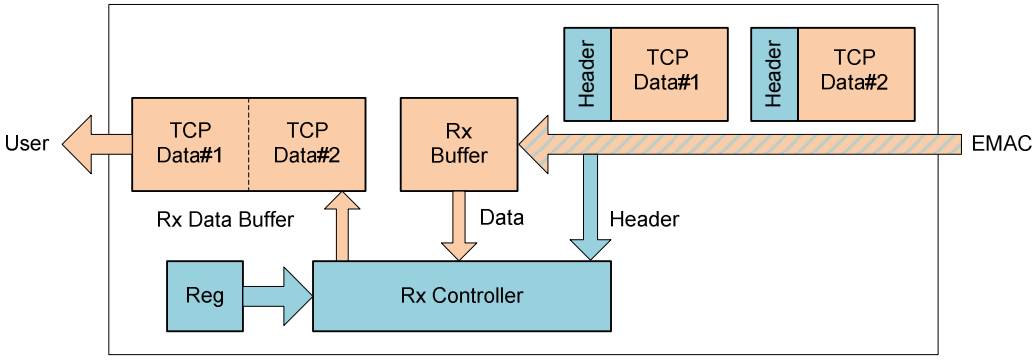

Figure 4: Receive Data Flow

When TOE40G-IP receives data from external device, Rx packet is stored to Rx buffer firstly to verify header and checksum. The network parameters of Rx packet must be matched to the set value in Reg. If parameter is not matched or checksum is error, the packet will be rejected. Data within valid packet is extracted and stored to Rx data buffer. The header of Rx packet is used to verify the packet, but does not store to Rx data buffer.

After valid packet is received, TOE40G-IP creates ACK packet to send data position which could be received completely. If data position of received packet is not arranged in correct sequence or lost packet is found, TOE40G-IP will generate duplicate ACK to request lost packet. IP busy flag is asserted to '1' during generating ACK packet.

However, TOE40G-IP can reorder the received packet sequence when packet sequence is swapped only one packet. For example, when received packet sequence is packet#1, #3, #2, and #4, TOE40G-IP can rearrange the data in correct sequence (packet#1, #2, #3, and #4). If packet is swapped more than one packet (such as packet#1, #3, #4, and #2), TOE40G-IP will generates duplicate ACK to request packet#2 after receiving #3 and #4.

Functional Description

TOE40G-IP core can be divided into three parts, i.e. control block, transmit block, and received block.

Control Block

- **Reg**

  User sets parameters for TCP/IP operation by using register interface. Register address is 4-bit, so maximum register is equal to 16. The description of each register is defined as shown in Table 2. After system reset is released to '0', all parameters are loaded to internal signals.

- **TCP Stack**

  TCP stack includes state machine to control the sequence to send and receive data following TCP/IP protocol. To send data, network parameters and command parameters such as total transmit size, packet size, and start data address of Tx data buffer are programmed to Transmit block by TCP stack. After that, TCP stack monitors Transmit block status until it completes to send all data.

  To receive data, TCP stack monitors received packet type returned by Received block. If it needs to genearate acknowledge packet, TCP stack will send the request to Transmit block to return the packet.

Table 2: Register map Definition

RegAddr [3:0]	Reg Name	Dir	Bit	Description
0000b	RST	Wr /Rd	[0]	Reset IP. '0': Normal operation, '1': Reset. Default value is '1'. **After all parameters are assigned, user sets '0' to this register to load parameter and start system intialization. Reset needs to be set/clear again to reload parameter if user changes the value of SML, SMH, DIP, SIP, DPN, or SPN register.**
0001b	CMD	Wr	[1:0]	User command in active mode. "00": Send data, "10": Open connection (active), "11": Close connection (active), "01": Undefined. **Before setting this register to send command (active mode), user needs to check system busy flag that is equal to '0' by reading bit[0] of this register.** The command operation starts after user sets this register.
		Rd	[0]	System busy flag. '0': Idle, '1': System is busy.
			[3:1]	Current operation. "000": Send data, "001": Idle, "010": Active open connection, "011": Active close connection, "100": Receive data, "101": Initialization, "110": Passive open connection, "111": Passive close connection.
0010b	SML	Wr /Rd	[31:0]	Define 32-bit lower MAC address (bit [31:0]) for this IP. **User needs to set this register before clearing RST register='0'.**
0011b	SMH	Wr /Rd	[15:0]	Define 16-bit upper MAC address (bit [47:32]) for this IP. **User needs to set this register before clearing RST register='0'.**
0100b	DIP	Wr /Rd	[31:0]	Define 32-bit target IP address. **User needs to set this register before clearing RST register='0'.**
0101b	SIP	Wr /Rd	[31:0]	Define 32-bit IP address for this IP. **User needs to set this register before clearing RST register='0'.**
0110b	DPN	Wr /Rd	[15:0]	Define 16-bit target port number. Set when the connection is opend by the IP (active open). **User needs to set this register before clearing RST register='0' (if user uses active open connection).** Target port number is auto defined from the packet during passive open connection. The packet to open connection must have network parameters matching to TOE40G-IP parameters.
0111b	SPN	Wr /Rd	[15:0]	Define 16-bit port number for this IP. **User needs to set this register before clearing RST register='0'.**
1000b	TDL	Wr	[31:0]	Total Tx data length in byte unit, but the size must be aligned to 32-byte. So, bit[4:0] is ignored by the IP. Valid from 32-0xFFFFFFE0. **User needs to set this register before setting CMD register = "00" (Send data). This value is loaded when CMD register is set. User can prepare the new value for the next transmission after TOE40G-IP starts send command operation.** If the next send command uses the same length, this register will not need to set again. TOE40G-IP uses the value from the latest send command.
		Rd	[31:0]	Remaining transfer length in byte unit which still not transmit.

RegAddr [3:0]	Reg Name	Dir	Bit	Description
1001b	TMO	Wr	[31:0]	Define timeout value for waiting Rx packet during running the command. The counter runs by using Clk input. So, timer unit depends on Clk frequency. For example, if Clk frequency is equal to 200 MHz, timer unit will be equal to 5 ns. This value should be more than latency time in the network to generate timeout when packet is lost, not normal condition.
		Rd		[0]-Timeout from not receiving ARP reply packet. After timeout, IP resends ARP request until ARP reply is received. [1]-Timeout from not receiving SYN and ACK flag during active open operation. After timeout, IP resends SYN packet for 16 times and then sends FIN packet to close connection. [2]-Timeout from not receiving ACK flag during passive open operation. After timeout, IP resends SYN/ACK packet for 16 times and then sends FIN packet to close connection. [3]-Timeout from not receiving FIN and ACK flag during active close operation. After 1st timeout, IP sends RST packet to close connection. [4]-Timeout from not receiving ACK flag during passive close operation. After timeout, IP resends FIN/ACK packet for 16 times and then sends RST packet to close connection. [5]-Timeout from not receiving ACK flag after sending data packet. After timeout, IP resends previous data packet. [6]-Timeout from Rx packet lost. IP generates duplicate ACK to request data retransmission. [22]-Receive FIN packet but data sending still not complete. [23]-Rx packet ignored because of Rx data buffer full (fatal error) [21],[27]-Rx packet lost [30]-RST packet is received [31],[29:28],[26:24]-Internal test status
1010b	PKL	Wr /Rd	[15:0]	Data size of Tx packet in byte unit. The size must be aligned to 32-byte, so bit[4:0] is ignored by the IP. Valid from 32-8960. Default value is 1440 byte (maximum size with 32-byte alignment for non-jumbo frame). **This value must not be changed when send command operation still not complete (Busy='1'). If the next send command uses same packet size, user will not need to set this register. The IP loads the previous value from latch register.**
1011b	PSH	Wr /Rd	[1:0]	Sending mode setting when TOE40G-IP transmits the last data packet. [0]-Disable to retransmit packet. Default value is 0. '0' - Generate duplicate data packet for the last packet to accelerate ACK packet. '1' – Disable data duplication for the last packet [1]-Enable to set PSH flag in all transmitted packets. Default value is 0. '0' – PSH flag of all transmit packets is equal to '0'. '1' – PSH flag of all transmit packets is equal to '1'.

RegAddr[3:0]	Reg Name	Dir	Bit	Description
1100b	WIN	Wr /Rd	[5:0]	Threshold value in 1Kbyte unit for sending windows update packet. Default value is 0 (Not enable window update feature). Threshold value is the minimum different value between current received buffer size and the window size of the latest transmit packet in order to generate windows update packet. The different value is calculated by (current received buffer size – the window size of the latest packet). Assume that WIN="000001b" (1 Kbyte) and window size of the latest transmit packet is equal to 2 Kbyte. The IP sends windows update packet after user read data out from the IP more than 1 Kbyte. The window size in windows update packet is equal to 3 Kbyte (2 Kbyte which is previous free space + 1 Kbyte which is additional free space after user reading) .
1101b	ETL	Wr	[31:0]	Extended total Tx length in byte unit. The size must be aligned to 32-byte, so bit[4:0] is ignored by the IP. User sets this register during running CMD="00" to increase total Tx length transfer. So, user could set total transmit size to be more than 4 GB without re-sending the new command to IP. For example, assumed that the 1st value of TDL=4 GB. When remaining size is 1 GB, user sets ETL=2 GB to complete 6 GB transmitted data (4 GB + 2 GB). The caution point to use this register is as follows. 1) ETL register could be set when read value of TDL is not less than 128 Kbyte. 2) The set value of ETL must be less than (0xFFFFFFE0 – read value of TDL) to avoild data counter overflow (data counter size is 32-bit).
1110b	SRV	Wr /Rd	[0]	'0': Client mode. The IP sends ARP request to get Target MAC address from IP address after IP reset is deasserted. IP busy is deasserted to '0' when ARP reply is returned to IP. '1': Server mode. The IP waits ARP request from the Target to get Target MAC address after IP reset is desserted. Next, IP returns ARP reply to the target and IP busy is deasserted to '0'. Default value is '0' (Client mode) **Note: In server mode, after TOE40G-IP is reset, the Target needs to resend ARP request to TOE40G-IP. Without receiving ARR request, TOE40G-IP cannot complete IP initialization.**

Table 3: TxBuf/ RxBufBitWidth Parameter description

Value of BitWidth	Buffer Size	TxBufBitWidth	RxBufBitWidth
9	16kByte	Valid	Valid
10	32kByte	Valid	Valid
11	64kByte	Valid	Valid

Transmit Block

- **Tx Data Buffer**

This buffer size is set by "TxBufBitWidth" parameter of the IP. The valid value is 9-11 which is equal to the address size of 256-bit buffer, as shown in Table 3.

The buffer size should be at least two times of Tx packet size (set by PKL register). Transmit data from user is stored to this buffer. Data in buffer is flushed after the target returns acknowledge packet to send data pointer which is received completely. Data in this buffer is split into small packet and forwarded to Tx packet buffer.

This buffer size is effect to transmit performance. If buffer is big, IP will send data continuously without waiting acknowledge returned from the target. It reduces the effect of delay time from network devices.

If total data of Tx data buffer is more than the total transmit size (set by TDL register), remaining data will be available in the buffer for the next transfer. The data in the buffer is flushed when the connection is closed or reset is detected. If the data in the buffer is less than packet size, IP will wait additional data from user to send the complete packet.

- **Tx Packet Buffer**

This buffer size is 16 Kbyte to store one data packet. Data in Tx packet buffer is forwarded to EMAC when TCP stack sends the request and EMAC is ready to receive new data.

- **Packet Builder**

TCP packet consists of the header and the data. Packet builder receives network parameter from Reg module and prepares TCP header. Also, IP and TCP checksum are also calculated to generate complete TCP header. After that, the header from internal logic following by the data from Tx packet buffer is transmitted to EMAC.

Received Block

- **Rx Buffer**

This is temporary buffer to store all Rx packets from EMAC. Data without header of valid packet is forwarded from Rx buffer to Rx data buffer.

- **Packet Filtering**

This module is designed to compare the header of received packet. The parameter in the header must be matched to the network parameter (set in Reg module). Also, IP and TCP checksum are calculated in this module. Only the packet which parameters are matched and checksum is correct is forwarded to Packet splitter.

- **Packet Splitter**

This module is designed to split the data from the received packet to store to Rx data buffer. Also, if the data is duplicate, the data will be ignored.

- **Rx Data Buffer**

This buffer size is set by “RxBufBitWidth” parameter of the IP. The valid value is 9-11. Free size of this buffer is used to be received window size of TOE40G-IP. Setting bigger size of this buffer can increase received performance because the data source continues sending data without waiting the acknowledge packet returned from TOE40G-IP. It reduces the effect of delay time in network devices.

User Block

This block is user module for setting command and monitoring status through register interface. Also, it includes the module to write data to TxFIFO interface and read data from RxFIFO interface. Simple state machine can be designed to control register interface.

40G/50G Ethernet MAC + BASE-R

The reference design uses 40/50 Gb Ethernet Subsystem which consists of EMAC and PCS/PMA for BASE-R, provided by Xilinx. More details of the IP are provided in following website.
https://www.xilinx.com/products/intellectual-property/ef-di-50gemac.html

Core I/O Signals

Descriptions of all parameters and I/O signals are provided in Table 4 and Table 5. MAC Interface of the IP uses 256-bit FIFO standard as FWFT mode.

Table 4: Core Parameters

Name	Value	Description
TxBufBitWidth	9-11	Setting Tx Data buffer size. The value is the address bus size of this buffer.
RxBufBitWidth	9-11	Setting Rx Data buffer size. The value is the address bus size of this buffer.

Table 5: Core I/O Signals

Signal	Dir	Description
Common Interface Signal		
RstB	In	Reset IP core. Active Low.
Clk	In	Clock input. Clock frequency should be at least 200 MHz to achieve high performance.
User Interface		
RegAddr[3:0]	In	Register address bus to write and read data. Must hold the same value during RegWrEn='1'.
RegWrData[31:0]	In	Register write data bus. Must hold the same value during RegWrEn='1'.
RegWrEn	In	Register write enable pulse. Assert to '1' wih the valid RegAddr and RegWrData.
RegRdData[31:0]	Out	Register read data bus. Valid in the next clock after setting RegAddr.
ConnOn	Out	Connection status ('1': connection is opened, '0': connection is closed)
TimerInt	Out	Timer interrupt. Assert to '1' for 1 clock cycle when timeout is detected or fatal error is found. The cause of TimerInt is checked by TMO[6:0] and TMO[23].
RegDataA1[31:0]	Out	32-bit read value of CMD register (RegAddr=0001b). Bit[0] is busy flag of TOE40G-IP.
RegDataA8[31:0]	Out	32-bit read value of TDL register (RegAddr=1000b). Used to monitor remaining transfer size when ETL register is applied.
RegDataA9[31:0]	Out	32-bit read value of TMO register (RegAddr=1001b). Used to be interrupt status when timeout is found.
Tx Data Buffer Interface		
TCPTxFfFlush	Out	Assert to '1' when Tx data buffer in TOE40G-IP is reset. This signal is asserted to '1' when connection is closed or IP is reset.
TCPTxFfFull	Out	Assert to '1' when Tx data buffer is almost full. User needs to stop writing data within 4 clock cycles after this flag is asserted.
TCPTxFfWrEn	In	Assert to '1' to write data to Tx data buffer.
TCPTxFfWrData[255:0]	In	Write data bus input to Tx data buffer. Valid at the same clock as TCPTxFfWrEn='1'.
Rx Data Buffer Interface		
TCPRxFfFlush	Out	Assert to '1' when Rx data buffer in TOE40G-IP is reset. This signal is asserted to '1' when new connection is opened.
TCPRxFfRdCnt[10:0]	Out	Data counter of Rx data buffer to show total data in 256-bit unit.
TCPRxFfLastRdCnt[4:0]	Out	Remaining byte of last data in Rx data buffer when total data is not aligned to 32-byte.
TCPRxFfRdEmpty	Out	Empty flag of Rx data buffer. User needs to stop reading data immediately when this signal is asserted to '1'.
TCPRxFfRdEn	In	Assert to '1' to read data from Rx data buffer.
TCPRxFfRdData[255:0]	Out	Read data output from Rx data buffer. Valid in the next clock after TCPRxFfRdEn is asserted to '1'.

Signal	Dir	Description
Mac FIFO Interface (FWFT mode)		
MacTxFfWrCnt[15:0]	In	Write data counter of MacTxFIFO in 256-bit. This signal is used to check full condition. If FIFO counter is less than 16-bit, please fill '1' to upper bit.
MacTxFfWrData[255:0]	Out	256-bit write data to MacTxFIFO. Valid at the same clock as MacTxFfWrEn='1'.
MacTxFfWrEn	Out	Assert to '1' to write data to MacTxFIFO. This signal is asserted to '1' continuously to transmit one Tx packet.
MacTxLastByteEn[31:0]	Out	Data byte enable of the last data in each Tx packet. This signal is valid when MacTxEnd='1' and MacTxFfWrEn='1'. In IP core, this signal could be equal to two values, i.e. 0x0FFF_FFFF and 0x003F_FFFF.
MacTxSizeData[15:0]	Out	Total size of Tx packet in 256-bit unit. This value is stable during MacTxFfWrEn='1'. Bit[15:11] is always equal to 0.
MacTxEnd	Out	Assert to '1' to indicate the final data of each Tx packet.
MacRxFfEmpty	In	Empty flag of MacRxFIFO. Assert to '1' when there is no data within MacRxFIFO.
MacRxFfRdEn	Out	Assert to '1' to read data from MacRxFIFO. This signal could be asserted to '1' when MacRxFfEmpty='0'.
MacRxFfRdData[257:0]	In	Read data output from MacRxFIFO. This signal is valid at the same clock as MacRxFfRdEn='1'. So, MacRxFIFO must be designed to run as FWFT mode. The definition of each received data is as follows. [255:0]: 256-bit received data from EMAC. [256]: Assert to '1' to indicate the final data in each Rx packet from EMAC. [257]: '0'-normal packet, '1'-error packet. This bit must be asserted to '1' at the same time as bit[256]='1' which indicates final data.

Timing Diagram

IP Initialization

For initialization process after user releases RST register=’0’, TOE40G-IP could run in two modes depending on SRV register setting, i.e. Client mode and Server mode.

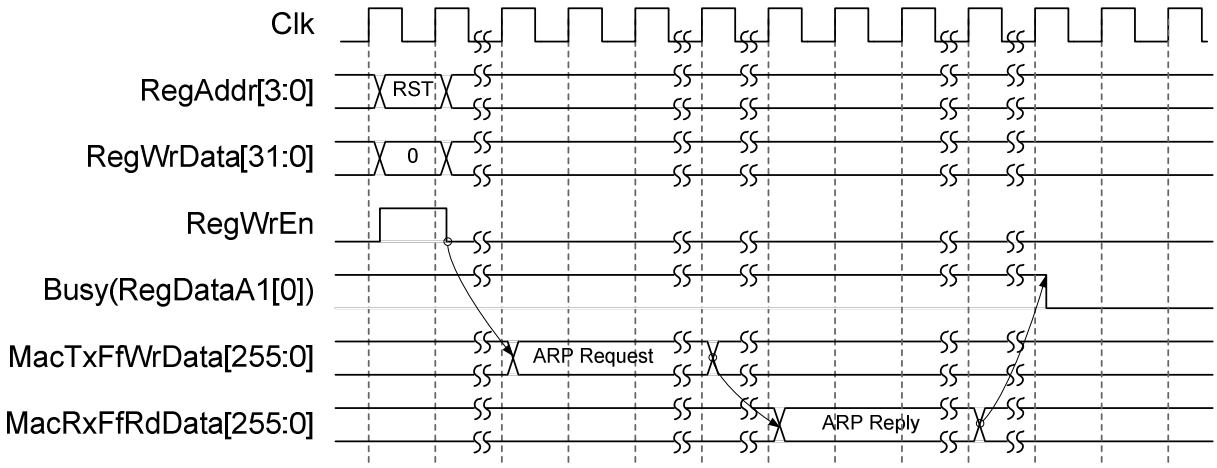

Figure 5: IP Initialization in Client mode

In Client mode, TOE40G-IP sends ARP request and waits ARP reply from the target. Target MAC address is extracted from ARP reply packet. After that, Busy signal is de-asserted to ‘0’. Busy signal could be monitored through bit10 of RegDataA1 output.

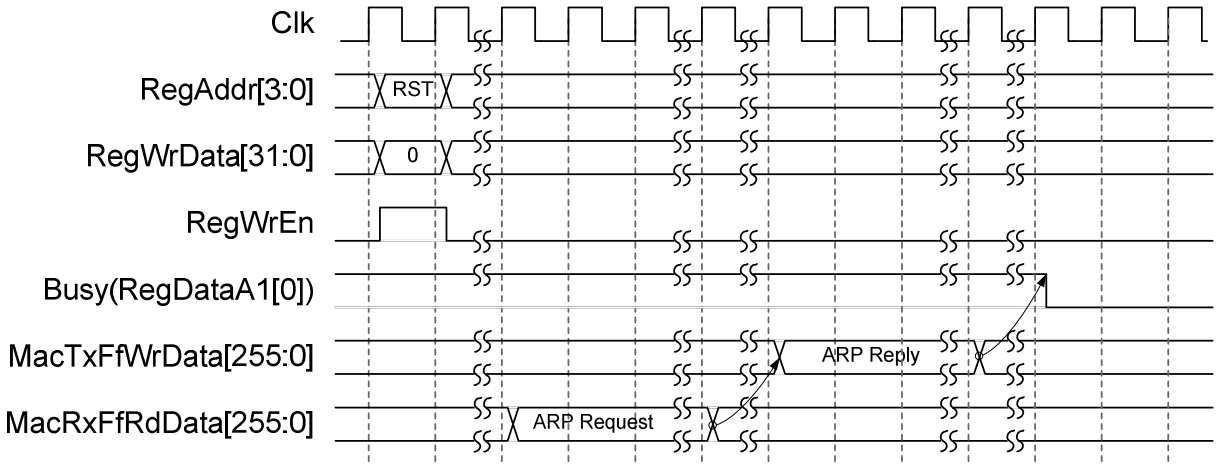

Figure 6: IP Initialization in Server mode

In Server mode, after TOE40G-IP reset is released to ‘0’, TOE40G-IP waits ARP request from the target. After receiving ARP request which the header is matched to network parameters of TOE40G-IP, TOE40G-IP returns ARP reply to the target. Target MAC address is extracted from ARP request packet. Finally, busy signal is de-asserted to ‘0’.

Register Interface

User can write/read control signals of TOE40G-IP by using Register interface. Timing diagram of register interface is shown in Figure 7. Register map is defined as shown in Table 2. To write register, user sets RegWrEn='1' with valid value of RegAddr and RegWrData. To read register, user sets RegAddr and then RegRdData is returned in the next clock.

Before user sets CMD register, busy flag (RegAddrA1[0] signal) must be monitored until it is equal to '0' (IP is in Idle status). After CMD register is set, busy flag is asserted to '1', as shown in Figure 8. Busy is de-asserted to '0' when the command is completed.

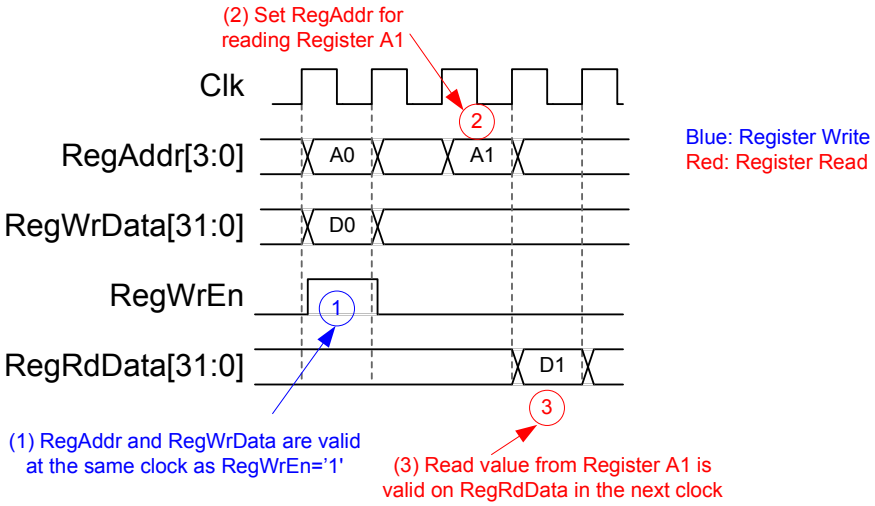

Figure 7: Register interface timing diagram

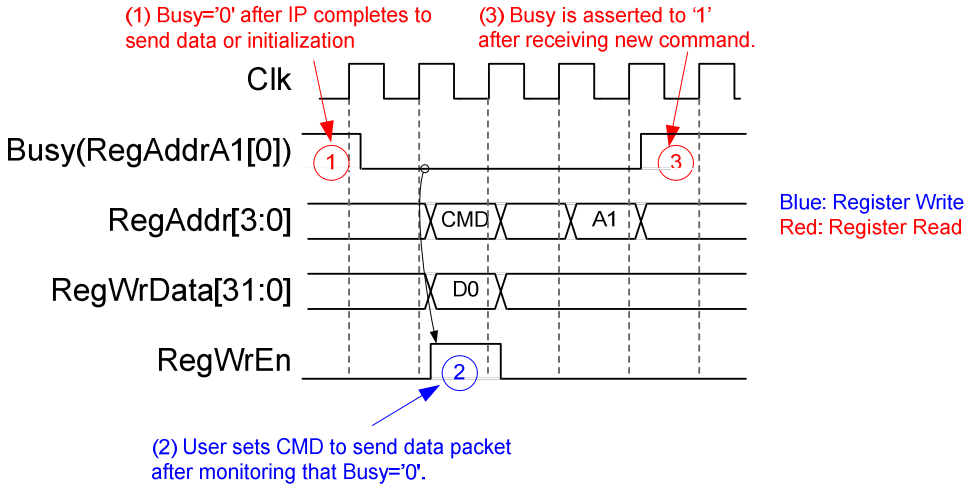

Figure 8: Set CMD register when busy is de-asserted

Tx FIFO Interface

User sends data to TOE40G-IP by using FIFO interface, as shown in Figure 9. Before sending data, user needs to check that full flag (TCPTxFfFull) is not asserted to '1' and ConnOn is equal to '1'. Next, set TCPTxFfWrEn='1' with valid TCPTxFfWrData. TCPTxFfWrEn must be cleared within 4 clock cycles to stop data sending after TCPTxFfFull is asserted to '1'. TCPTxFfFlush is asserted to '1' when data in Tx data buffer is flushed. TCPTxFfFlush could be asserted to '1' by two situations, i.e. connection closed and IP reset.

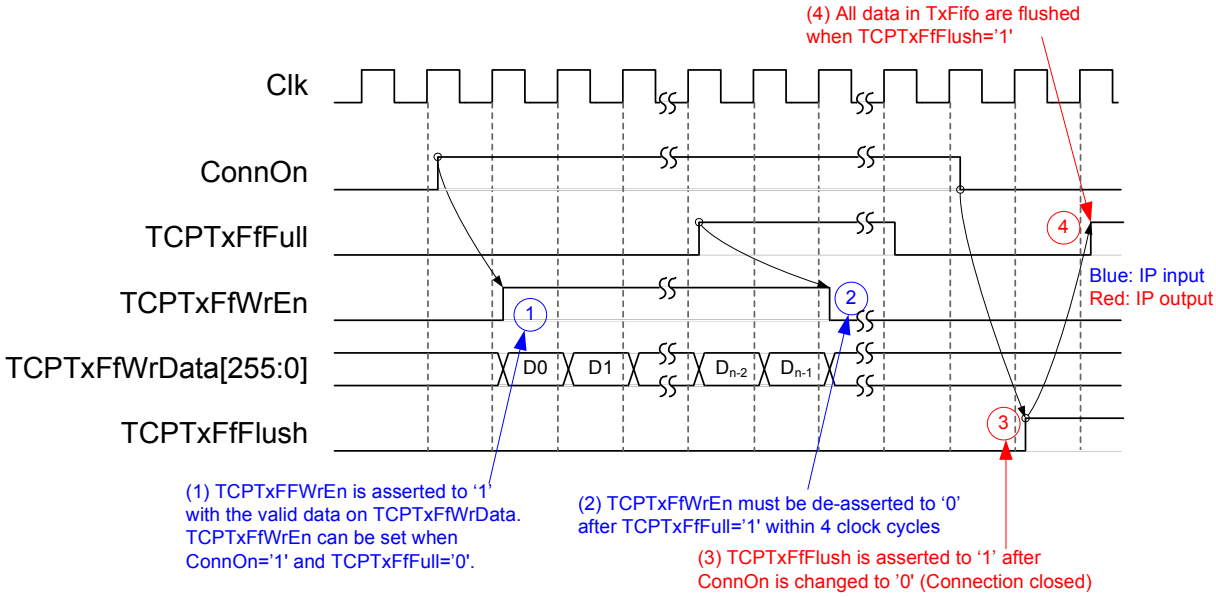

Figure 9: Tx data buffer interface timing diagram

Rx FIFO Interface

The received data extracted from valid packet is stored in Rx data buffer. User read data from Rx data buffer through RxFIFO interface, as shown in Figure 10. RxFIFO status is monitored through TCPRxFfEmpty signal (data can be read when TCPRxFfEmpty is cleared to '0'). TCPRxFfRdEn is asserted to '1' to read data from Rx data buffer and TCPRxFfRdData is valid in the next clock. Data reading must stop immediately by deasserting TCPRRxFfRdEn to '0' when TCPRxFfEmpty = '1'. All data in Rx data buffer are flushed when the new connection is opened (TCPRxFfFlush is also asserted to '1').

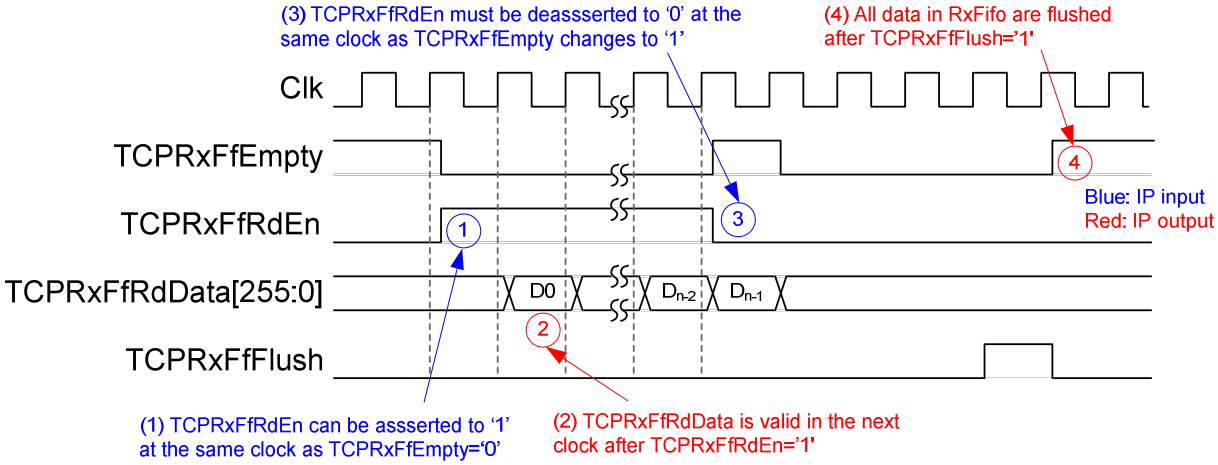

Figure 10: Read Rx data buffer by empty flag timing diagram

To read data as burst transfer, TCPRxFfRdCnt can be used to check total data inside Rx data buffer. User logic asserts TCPRxFfRdEn='1' for many clock cycles to read data from the buffer, as shown in Figure 11.

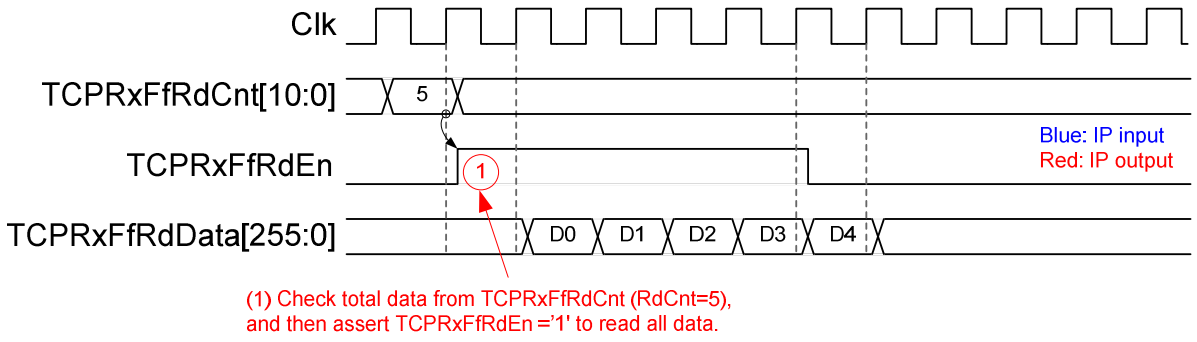

Figure 11: Read Rx data buffer by read counter timing diagram

MAC FIFO Interface

EMAC interface of TOE40G-IP is designed to be FIFO in FWFT mode interface. To integrate TOE40G-IP with 40 EMAC, adapter logic including FWFT FIFO is recommended as shown in Figure 12.

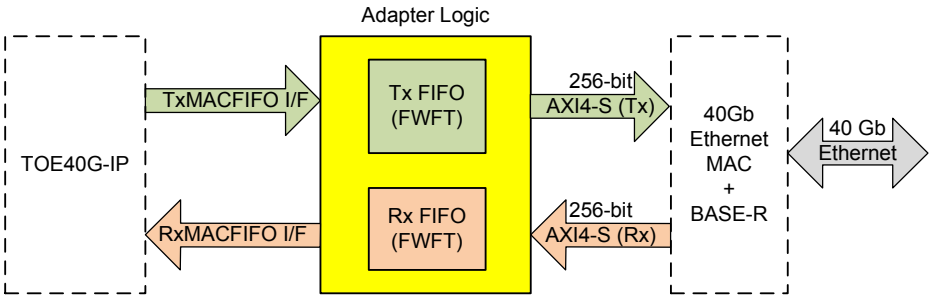

Figure 12: TOE40G-IP and EMAC connection

Before transmit packet, TOE40G-IP monitors MacTxFfWrCnt firstly to confirm free space inside MacTxFIFO which must be much enough to store one packet. After that, MacTxFfWrEn is asserted to '1' continuously until end of packet. MacTxFfWrData is valid at the same clock as MacTxFfWrEn='1'. At the end of packet, MacTxEnd is asserted to '1' with the valid MacTxLastByteEn. MacTxSizeData does not change during transmitting each packet to show total data of the packet in 256-bit unit. The example of timing diagram when TOE40G-IP sends packet (packet size = n) is shown in Figure 13.

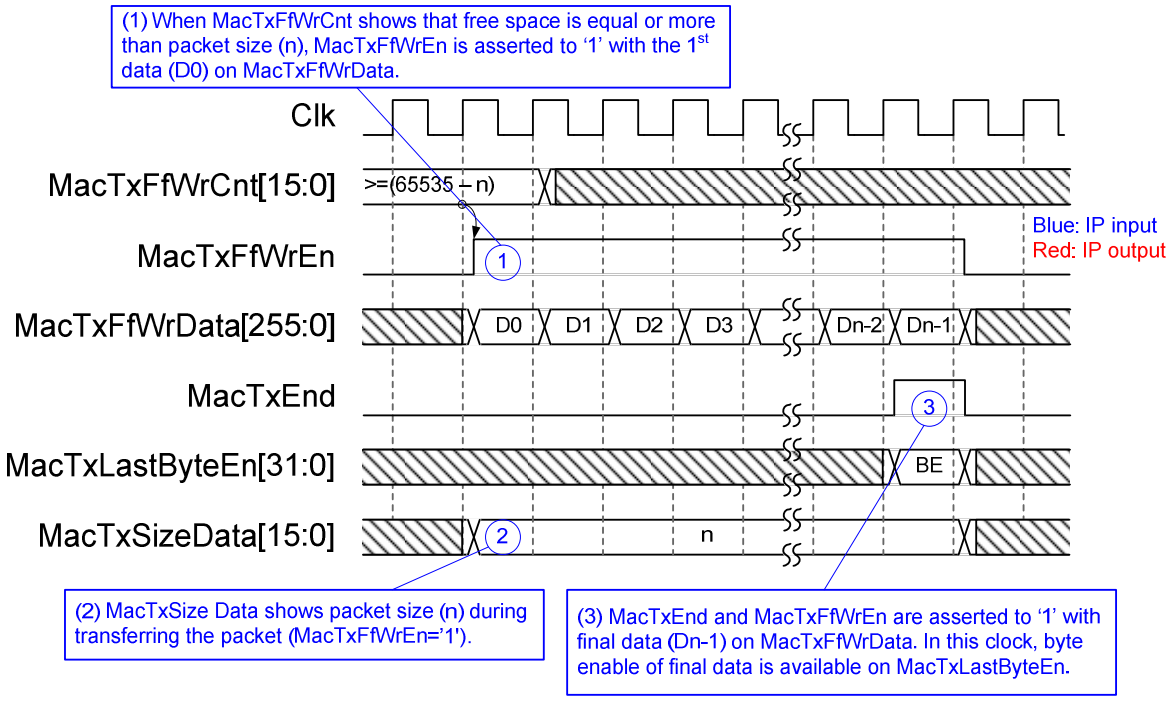

Figure 13: MacTxFIFO interface timing diagram

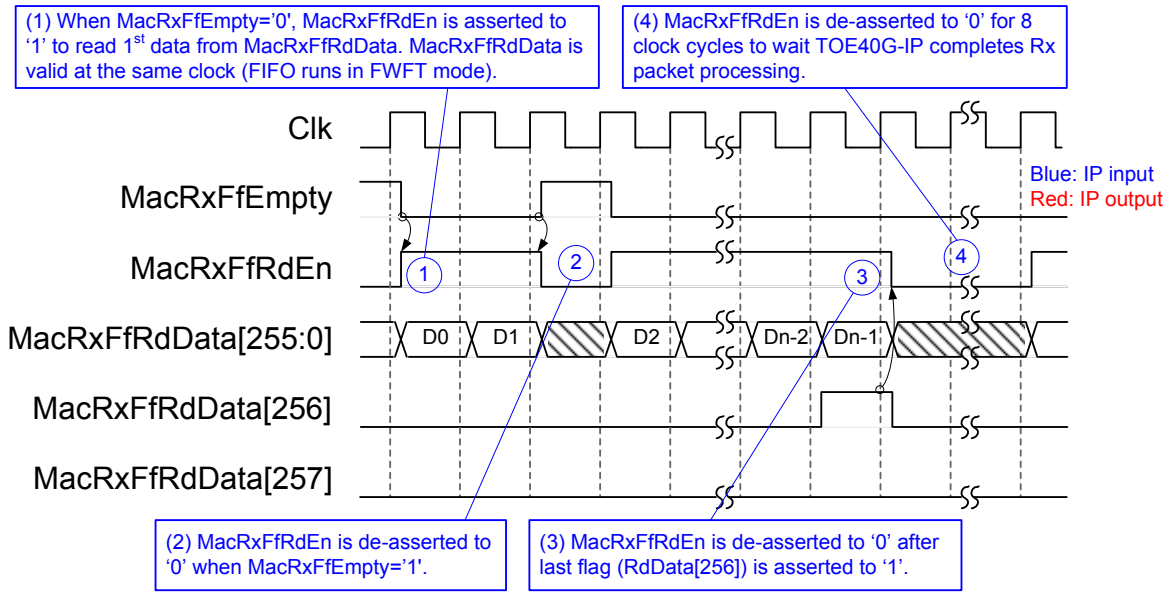

Figure 14: MacRxFIFO interface timing diagram

Figure 14 shows the example of MacRxFIFO interface when Rx packet size is equal to n. MacRxFfEmpty is applied to monitor data status in MacRxFIFO. MacRxFfRdEn is asserted to '1' to read data from MacRxFIFO when MacRxFfEmpty='0'. According to FWFT characteristic, MacRxFfRdData is available to read at the same clock as MacRxFfRdEn='1'. Bit 256 of MacRxFfRdData is asserted to '1' when current data is the final data of the packet. After receiving final data, MacRxFfRdEn is de-asserted to '0' for 8 clock cycles. 8-clock gap is the time to wait TOE40G-IP completes received packet processing. After that, MacRxFfRdEn is asserted to '1' (if MacRxFfEmpty='0') to start new packet processing.

From above timing diagram, adapter logic must be designed to convert MacFIFO interface on TOE40G-IP to 256-bit AXI4 stream interface on 40Gb EMAC. The example of adapter logic is included in TOE40G-IP reference design.

Example usage

Client mode (SRV[0]='0')

The example of the sequence to set register for data transmission and reception in Client mode is shown as follows.

1) Set RST register='1' to reset the IP.
2) Set SML/SMH for MAC address, DIP/SIP for IP address, and DPN/SPN for port number. (DPN is optional setting when the port is opened by TOE40G-IP or active open).
3) Set RST register='0' to clear the reset and then IP starts initialization by sending ARP request to get Target MAC address from ARP reply. After end of initialization, busy signal (RegAddrA1[0]) is de-asserted to '0'.
4) The new connection is created by two modes.

   a. Active open: user sets CMD register to create the connection (the 1st SYN packet is sent from TOE40G-IP).

   b. Passive open: wait until "ConnOn" signal = '1' (the 1st SYN packet is sent from the target).
5) a. For data transmission, user sets TDL (total Tx data) and PKL (Tx packet size). Next, user sets CMD register = send data and then sends data to TOE40G-IP through TxFIFO interface. User waits until send command complete (Busy flag changes to'0'). For next send command, user sets TDL, PKL, and CMD register without IP reset.

   b. For data reception, user monitors received data through RxFIFO interface. When new data is received, user reads data until RxFIFO is empty.
6) Similar to create the connection, the connection is destroyed by two modes.

   a. Active close: user sets CMD register to close the connection (FIN packet is 1st sent from TOE40G-IP).

   b. Passive close: wait until "ConnOn" signal = '0' (FIN packet is 1st sent from the target).

Server mode (SRV[0]='1')

The different point between Server mode and Client mode is the initialization process to get MAC address of the target. In Client mode, MAC address is received from ARP reply after TOE40G-IP sends ARP request. In Server mode, MAC address is decoded from ARP request which has matched Target IP address. The process to send and receive data is same as Client mode. The example sequence of Server mode is shown as follows.

1) Set RST register='1' to reset the IP.
2) Set SML/SMH for MAC address, DIP/SIP for IP address, and DPN/SPN for port number.
3) Set RST register='0' to clear the reset. IP starts initialization by waiting ARP request to get Target MAC address. Next, IP creates ARP reply to the Target. After end of intialization, busy signal is de-asserted to '0'.
4) Remaining steps are similar to step 4 – 6 of Client mode

Verification Methods

The TOE40G-IP Core functionality was verified by simulation and also proved on real board design by using ZCU102/ZCU106 evaluation board.

Recommended Design Experience

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

Ordering Information

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

Revision History

Revision	Date	Description
1.0	Oct-3-2018	New release