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 AC701, KC705, VC707, ZC706, Zynq Mini-ITX(Z100)
Simulation Tool Used	
ModelSim	
Support	
Support Provided by Design Gateway Co., Ltd.	

Features

- TCP/IP stack implementation
- Support IPv4 protocol
- Support one session per one TOE1G-IP (Multisession can be implemented by using multiple TOE1G-IP)
- Support both Server and Client mode (Passive/Active open and close)
- Transmit/Receive buffer size, adjustable for optimized resource and performance
- Simple data interface by standard FIFO interface
- Simple control interface by standard register interface
- One clock domain interface by fixed 125 MHz clock frequency
- Reference designs available on AC701/KC705/VC707/ZC706/Zynq Mini-ITX(Z100) board, i.e. half-duplex, full-duplex, ftp server, and two-port reference design
- Jumbo frame support as optional
- Not support data fragmentation feature

Table 1: Example Implementation Statistics

Family	Example Device	Fmax (MHz)	Slice Regs	Slice LUTs	Slices [1]	IOB [2]	RAMB36E1	RAMB18E1	Design Tools
Artix-7	XC7A200T-2FBG676	125	2760	2633	963	136	36	3	Vivado2015.4
Kintex-7	XC7K325T-2FFG900	125	2760	2634	1013	136	36	3	Vivado2015.4
Zynq-7000	XC7Z045-2FFG900	125	2760	2635	997	136	36	3	Vivado2015.4
Virtex-7	XC7VX485T-2FFG1761	125	2760	2632	987	136	36	3	Vivado2015.4

Notes:

1) Actual logic resource dependent on percentage of unrelated logic

2) Assuming all core I/Os and clocks are routed off-chip

3) Block memory resources are based on 64k Tx data buffer size, 16k Tx packet buffer size, and 64k Rx data buffer size. Minimum size of each buffer are 4k Tx data buffer size, 2k Tx packet buffer size, and 2k Rx data buffer size.

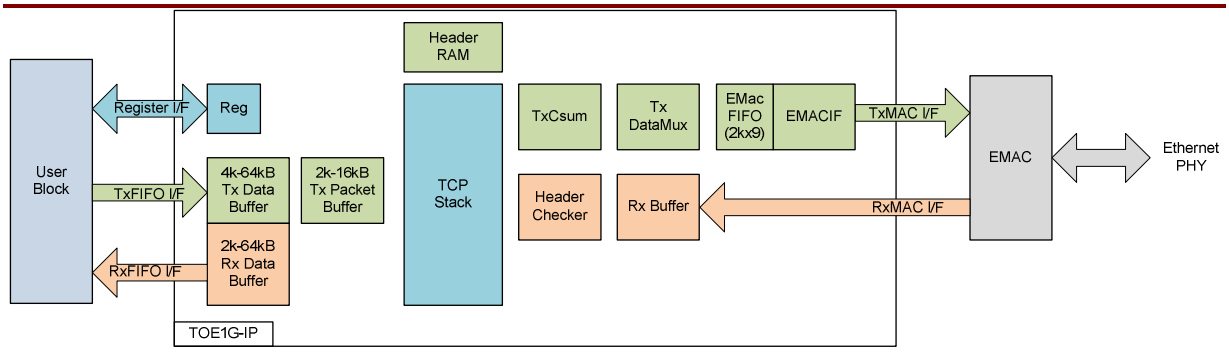

Figure 1: TOE1G-IP Block Diagram

Applications

TOE1G-IP is designed for network application by using TCP/IP protocol with high speed bandwidth transfer such as network data storage, IP camera, Printer server. By using this IP, user can easily transfer data with any device through TCP/IP protocol without CPU usage in system.

General Description

TOE1G-IP core operating with Xilinx EMAC IP core can operate TCP/IP stack, Transport layer, Internet layer, and Link layer for network data transmission. User can send and receive data with any network device through TCP/IP protocol by using this system and external PHY chip.

There are three types of user interface, i.e. control signal by register access, transmit and received data signal by FIFO access. During initializing system, user needs to set up system parameter such as packet size, port number, IP number through register interface. Packet data transferring can be operated by two modes, i.e. Active mode and Passive mode. On Active mode, user can control through register interface for opening connection, closing connection, and sending data from Tx Data buffer to external network device. On Passive mode, port connection will be opened, closed from external device. Also, data from external device will be stored to Rx data buffer within TOE1G-IP.

The size of three buffers in TOE1G-IP (Tx Data buffer, Tx Packet buffer, and Rx Data buffer) can be selected by setting parameter of the IP. The different size is provided to optimize resource utilization for user application. The bigger size takes much resource, but achieve the better transfer performance. Tx Data buffer size and Tx Packet buffer size are effect to transmit performance, while Rx Data buffer is effect to receive performance. Otherwise, Tx Data buffer size and Tx Packet buffer size are related to the packet size which user can be programmed through register interface. Tx Packet buffersize must be more than the Tx packet size while Tx Data Buffer size should be at least two times of the Tx packet size.

To transmit data, data from Tx Data buffer will be split into packet size and then fed to Tx Packet buffer. Data output from Tx Packet buffer will be combined with header data in Header RAM before sending out to EMAC. TCP and IP checksum will be auto calculated within TOE1G-IP. Acknowledge number of Rx packet will be monitored. In normal condition, acknowledge number will be updated to show that the buffer inside the target is available to receive additional packet, so IP will send next data packet. But if acknowledge number is same as previous packet, IP will detect duplicate ACK condition and retransmit data packet in Tx buffer. Busy flag (monitored from register access or IP port output) will be cleared after completed data transfer size is equal to setting value from user. User can monitor this busy flag to check transfer status.

For receiving data, Rx packet will be stored to temp buffer firstly. Header and checksum within Rx packet will be verified. If header or checksum is error, the packet will be rejected and not store to Rx Data buffer. When correct data packet is received, data will be stored to Rx Data buffer and Acknowledge packet will be sent out from TOE1G-IP to request more data packet from external network device. If data lost is found, IP will generate duplicate packet to request data retransmission. TOE1G-IP will go back to Idle state (Busy flag='0') when no more packet is received and the sequence of received packet is correct.

User can change packet size and total transfer size for new transmit without closing the port if the IP is Idle state.

Functional Description

TOE1G-IP core can be divided into three parts, i.e. control block, transmit block, and received block.

Control Block

- **Reg**

  User can set parameters for TCP/IP operation by using register interface. Register address of this interface is equal to 4-bit. The description of each register address is defined as shown in Table 2. After system reset is released, all internal parameters will be updated by setting value of each register.

- **TCP Stack**

  To operate active command from user, TCP Stack will decode user command and start transmit block to transmit packet out. After sending packet, TCP Stack will monitor received packet from received block to check the Acknowledge packet.

  To operate passive command from external device, TCP Stack receives packet from received block, and then decode it. After that, TCP Stack will start transmit block to generate the Acknowledge packet.

Table 2: Register map Definition

RegAddr [3:0]	Reg Name	Dir	Bit	Description
0000b	RST	Wr /Rd	[0]	Reset IP. '0': Release reset, '1': Reset. Default value is '1'. **After user setting all parameters, set '0' to this register to release reset and start system parameter intialization. Reset needs to be set again if user will change 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 start any active command, user needs to check system busy flag by reading bit[0] of this register to confirm that there is no operation running.** Active command will auto-start after this register is set by user.
		Rd	[0]	System busy flag. '0': Idle, '1': System operating.
			[3:1]	Current operation. "000": Send data with/without received data, "001": Idle, "010": Active open connection, "011": Active close connection, "100": Receive data, "101": Undefined, "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.**
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.**
0100b	DIP	Wr /Rd	[31:0]	Define 32-bit target IP address. **User needs to set this register before clearing RST register.**
0101b	SIP	Wr /Rd	[31:0]	Define 32-bit IP address for this IP. **User needs to set this register before clearing RST register.**
0110b	DPN	Wr /Rd	[15:0]	Define 16-bit target port number. **User needs to set this register before clearing RST register if user wants to use active open connection.** Target port number will be auto defined from passive open connection packet which parameters in header are matched with setting value.
0111b	SPN	Wr /Rd	[15:0]	Define 16-bit port number for this IP. **User needs to set this register before clearing RST register.**
1000b	TDL	Wr	[31:0]	Total Tx data length transfer in byte unit. Valid from 1-0xFFFFFFFF. **User needs to set this register before setting CMD register = "00". This value will be latched to internal logic when CMD register is set. So, user can prepare the new value for next transmit after setting CMD register.** If user will transmit data with same length, this register doesn't need to set again. Previous value will be used from internal latch.
		Rd	[31:0]	Remaining data 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 any command. This register is used by 125 MHz counter, so timer unit is about 8 ns. This value must be more than 0x6000.
		Rd		[0]-Timeout from not receiving ARP reply packet After timeout, IP will resend ARP request until ARP reply is received. [1]-Timeout from not receiving SYN and ACK flag during active open operation After timeout, IP will resend SYN packet for 16 times and then send FIN packet to close connection. [2]-Timeout from not receiving ACK flag during passive open operation After timeout, IP will resend SYN/ACK packet for 16 times and then send FIN packet to close connection. [3]-Timeout from not receiving FIN and ACK flag during active close operation After timeout, IP will send RST packet to close connection. [4]-Timeout from not receiving ACK flag during passive close operation After timeout, IP will resend FIN/ACK packet for 16 times and then send RST packet to close connection. [5]-Timeout from not receiving ACK flag during data transmit operation After timeout, IP will resend previous data packet. [6]-Timeout from Rx packet lost, Rx data FIFO full, or wrong sequence number IP will generate duplicate ACK to request data retransmission. [23]-Rx packet ignored because of Rx data buffer full [27]-Rx packet lost detected [30]-RST flag is detected in Rx packet [31],[29:28],[26:24]-Internal test status
1010b	PKL	Wr /Rd	[15:0]	Data length of Tx packet in byte unit. Valid from 1-16000. Default value is 1460 byte (Maximum size for non-jumbo frame). **This value must not be changed during data transmission not complete (Busy='1'). If next transmit still use same packet size, user does not need to set this register because the previous value is latched in the logic.**
1011b	PSH	Wr /Rd	[1:0]	Sending mode setting when TOE1G-IP transmits only one packet (PKL = TDL). [0]-Disable to retransmit packet. Set '0' to generate duplicate data packet. (Default = '0'). [1]-Enable to set PSH flag in transmit packet. Set '1' to insert PSH flag in TCP header. (Default = '0')
1100b	WIN	Wr	[5:0]	Threshold value in 1Kbyte unit for transmit windows update packet. Default value is 0 (Not enable window update feature). The IP will transmit windows update packet when received buffer size is now changed from the latest transmit packet more than threshold value. For example, if WIN="000001b" or 1 Kbyte and window size of the latest transmit packet (the latest size of received buffer in the IP) is equal to 2 Kbyte. After that, user read data out from the IP more than 1 Kbyte. The IP will send Windows update packet to update window size to be 3 Kbyte.

Note:

1. Target Mac address is defined from returned value in ARP Reply packet, so user doesn't need to set this parameter.
2. Target Port number is defined from received packet when the port is opened in passive mode.

Table 3 TxBuf/TxPac/RxBufBitWidth Parameter description

Value of BitWidth	Buffer Size	TxBufBitWidth	TxPacBitWidth	RxBufBitWidth
11	2kByte	No	Valid	Valid
12	4kByte	Valid	Valid	Valid
13	8kByte	Valid	Valid	Valid
14	16kByte	Valid	Valid	Valid
15	32kByte	Valid	No	Valid
16	64kByte	Valid	No	Valid

Transmit Block

- **Tx Data Buffer**

This buffer size is set by "TxBufBitWidth" parameter of the IP. The valid value is 12-16 which is equal to the address size of buffer, as shown in Table 3.

The buffer size should be at least two times of Tx Packet Size in PKL register. Transmit data from user will be stored within this buffer, and flushed after the target returns acknowledge packet to confirm that data is received completely. Data from this buffer is read out to calculate checksum of each packet before storing to Tx Packet Buffer.

This buffer size is effect to the total performance. If the size is much enough, IP can send data out continuously without waiting acknowledge returned from the target. So, data latency from all processes and the carrier will not be effect to the performance.

If user sends data more than the total transmit size, remaining data will be available in the buffer for next transfer. The data in buffer will be flushed when the port is closed or reset is detected. If the data in the buffer is not enough for the current transaction, IP will not send out the packet and wait data from user.

- **Tx Packet Buffer**

The size is set by "TxPacBitWidth" parameter of the IP. The valid value is 11-14 and the description of the parameter is shown in Table 3. This buffer size must be more than to Tx Packet size (setting in PKL register) to store at least one packet data splitting from Tx data buffer. Maximum value of PKL register is equal to (Tx Packet Buffer size – 4). Data in Tx Packet buffer is sent out when EMAC and target ready to receive data. During sending current packet, next packet will be prepared by receiving new data from Tx Data buffer, so packet can be sent out continuously.

- **Header RAM**

This RAM is applied to store header part of Transmit packet including checksum value. The main parameters in the packet are programmed by register after user release RST register. Some parameters such as Target MAC address and Target port number can be updated by ARP Reply or Passive open packet.

- **TxCsum**

This module is desigend to calculate checksum of each Tx packet before sending out. The checksum output will be stored within Header RAM.

- **TxDataMux**

This module is desigend to merge header from Header RAM and data from Tx Packet buffer into one packet and send out to EMAC.

- **EMACIF and EMacFIFO**

EMacFIFO is 2k x 9-bit FIFO to support flow control of transmit data to EMAC.

Received Block

- **Rx Buffer**

This is temporary buffer to store all Rx packets from EMAC. The buffer is used to wait Header Checker processing that current packet is whether valid or not. Only valid TCP data will be flushed from Rx buffer to store to Rx Data buffer.

- **Header Checker**

Header in Rx packet will be checked and compared with parameter in register. Packet within Rx Buffer will be ignored if any parameter is not matched or checksum is error. Also, if duplicate data is detected in valid packet, the data in the newer packet will be ignored and not transferred to Rx Data buffer.

- **Rx Data Buffer**

This buffer size is set by “RxBufBitWidth” parameter of the IP. The valid value is 11-16 and the description of the parameter is shown in Table 3. This buffer size is used to be received window size of this TCP connection. Setting bigger size to this buffer can increase received data performance because the data source can continue sending data without waiting the acknowledge returned from TOE1G-IP which may be delayed from network routing, the process within the data source, or received buffer full. Also, using big buffer can help TOE1G-IP to store the received packet and rearrange data when the received packet sequence is swapped from network routing.

User Block

This block is user module for setting and monitoring register interface, writing data to Tx FIFO, and reading data from Rx FIFO. This module can be designed by simple hardware logic.

EMAC

The reference design of the IP uses TEMAC core from Xilinx. More details of the IP are provided in following website.
https://www.xilinx.com/products/intellectual-property/temac.html

Core I/O Signals

Descriptions of all parameters and signal I/O are provided in Table 4 and Table 5. All signals in MAC Interface group are designed to connect to Xilinx EMAC port directly.

Table 4: Core Parameters

Name	Value	Description
TxBufBitWidth	12-16	Setting Tx Data buffer size. The value is referred to address bus size of this buffer.
TxPacBitWidth	11-14	Setting Tx Packet buffer size. The value is referred to address bus size of this buffer.
RxBufBitWidth	11-16	Setting Rx Data buffer size. The value is referred to address bus size of this buffer.

Table 5: Core I/O Signals

Signal	Dir	Clk	Description
Common Interface Signal			
RstB	In		Reset IP core. Active Low.
Clk	In		125 MHz fixed clock frequency input for user interface and MAC transmit interface for 1 Gbps mode.
User Interface			
RegAddr[3:0]	In	Clk	Register address bus
RegWrData[31:0]	In	Clk	Register Write data bus
RegWrEn	In	Clk	Register Write enable pulse. Assert with valid value of RegAddr and RegWrData signals.
RegRdData[31:0]	Out	Clk	Register Read data bus. Available after asserting RegAddr with 1 Clk period latency
ConnOn	Out	Clk	Connection Status (‘1’ : connection is opened, ‘0’: connection is closed)
TimerInt	Out	Clk	Timer interrupt. Assert to high for 1 Clk period when time out is detected. User can read TMO[6:0] register to check interrupt status.
Busy	Out	Clk	IP busy status (‘0’: IP is idle, ‘1’: IP is busy). This signal is same as CMD register bit 0.
Tx Data Buffer Interface			
TCPTxFfFlush	Out	Clk	Transmit buffer within IP is reset. Assert to high only 1 Clk period when connection is closed or reset.
TCPTxFfFull	Out	Clk	Transmit buffer full flag. User needs to stop writing data within 4 clock period after this flag is asserted to high.
TCPTxFfWrEn	In	Clk	Transmit buffer write enable. Assert to write data to Transmit buffer.
TCPTxFfWrData[7:0]	In	Clk	Transmit buffer write data bus. Synchronous with TCPTxFfWrEn.
Rx Data Buffer Interface			
TCPRxFfFlush	Out	Clk	Received buffer within IP is reset. Assert to high only 1 Clk period when connection is opened.
TCPRxFfRdCnt[15:0]	Out	Clk	Received buffer data counter to show total number of received data in buffer.
TCPRxFfRdEmpty	Out	Clk	Received buffer empty flag. User needs to stop reading data immediately.
TCPRxFfRdEn	In	Clk	Received buffer read enable. Assert to read data from Received buffer.
TCPRxFfRdData[7:0]	Out	Clk	Received buffer read data bus. Valid after TCPRxFfRdEn assert with 1 Clk period latency.

Signal	Dir	Clk	Description
MAC Interface			
MacRxClk	In		Received clock from EMAC core.
MacRxReset	In		Active-High Rx software reset from EMAC core. This signal is unused.
MacRxData[7:0]	In	RxClk	Received data bus from EMAC core.
MacRxValid	In	RxClk	Received data valid signal from EMAC. Synchronous with MacRxData.
MacRxLast	In	RxClk	Control signal to indicate the final byte in the frame.
MacRxUser	In	RxClk	Control signal asserted at the end of received frame to indicate that the frame has an error. '0': normal packet, '1': error packet.
MacTxReset	In		Active-High Tx software reset from EMAC core. This signal is unused.
MacTxData[7:0]	Out	Clk	Transmitted data to EMAC core.
MacTxValid	Out	Clk	Transmitted data valid signal to EMAC. Synchronous with MacTxData.
MacTxLast	Out	Clk	Control signal to indicate the final byte in a frame.
MacTxUser	Out	Clk	Control signal to indicate an error condition. This signal is always '0'.
MacTxReady	In	Clk	Handshaking signal. Asserted when MacTxData has been accepted.

Timing Diagram

Register Interface

User can write/read control signal with TOE1G-IP by using Register interface which has timing diagram as shown in Figure 2. Register map address is designed as shown in Table 2. To write control signal, User needs to set RegWrEn='1' with valid value of RegAddr and RegWrData. To read control signal, User set only RegAddr value and then RegRdData will be valid in next clock period.

Before user set CMD register, Busy flag must be monitored from pin or register access to confirm that IP is in Idle status. After CMD register is set, Busy flag will be asserted to '1' to show that IP start the operation, as shown in Figure 3.

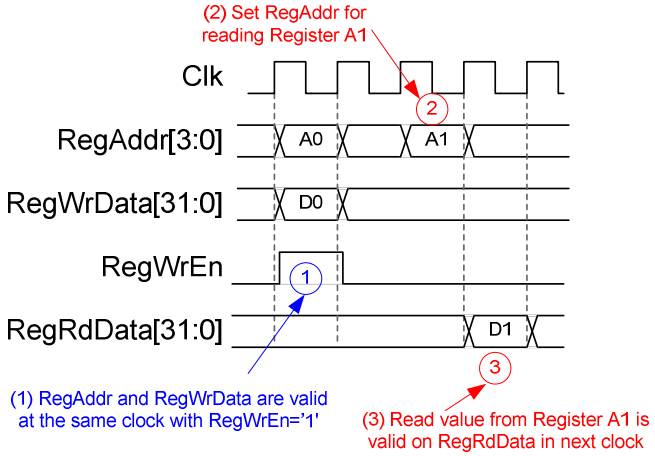

Figure 2: Register Interface Timing Diagram

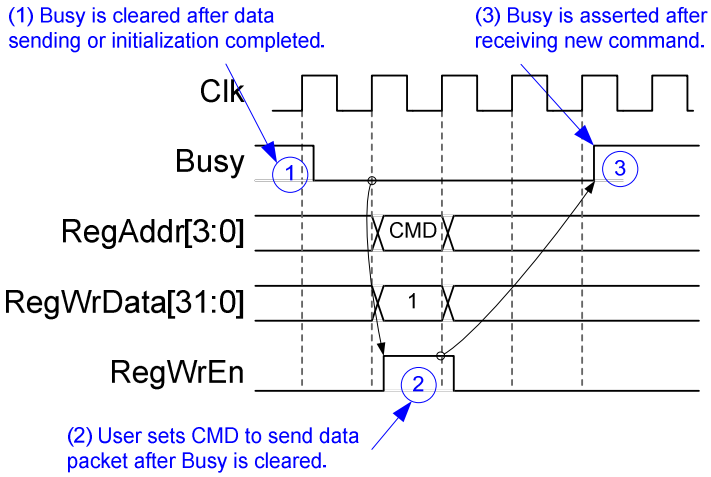

Figure 3: Set CMD register when Busy is de-asserted

Tx FIFO Interface

User can send data to IP core by using FIFO interface, as shown in Figure 4. Before sending data, user needs to check full flag (TCPTxFfFull) that is not asserted to '1' and ConnOn is equal to '1'. Then, set TCPTxFfWrEn='1' with valid value of TCPTxFfWrData. TCPTxFfWrEn must be cleared within 4 clock period to stop data sending after TCPTxFfFull is asserted to '1'. TCPTxFfFlush will be asserted to '1' from IP core to inform user that all data in Tx FIFO are cleared which is caused from close connection detect.

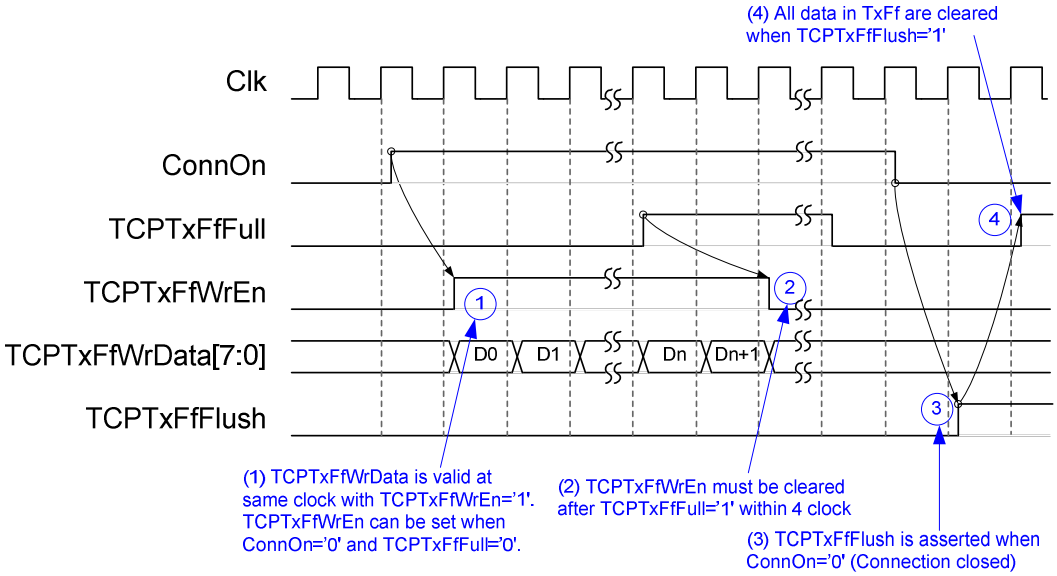

Figure 4: Tx Data Buffer Interface Timing Diagram

In normal case, TCPTxFfFull flag can be negated from ‘1’ to ‘0’ by two conditions.

1) After change RST register from ‘1’ to ‘0’ to release internal IP reset and start system parameter initialization. During IP reset, TxBuffer inside the IP will be also reset and data will be flush, so Full flag is asserted to block data input from user.

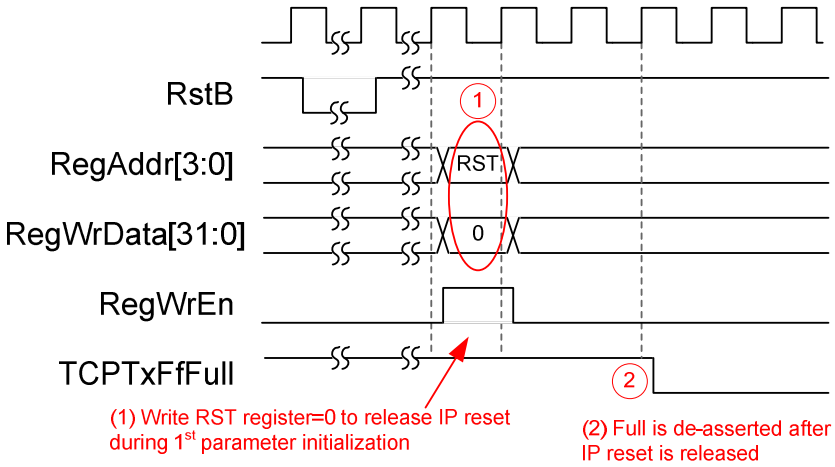

Figure 5: TCPTxFfFull de-asserted from IP reset is released

2) After ConnOn=’1’ and CMD register still not set to Send data mode. The read pointer of TxBuffer is related to the acknowledge number of returned ACK packet. When port is opended (ConnOn=’1’), the acknowledge number and read pointer will be updated which is effect to. TCPTxFfFull asserted. The write pointer of TxBuffer will be updated when user sets CMD register=Send Data, so TCPTxFfFull will be de-asserted, as shown in Figure 5.

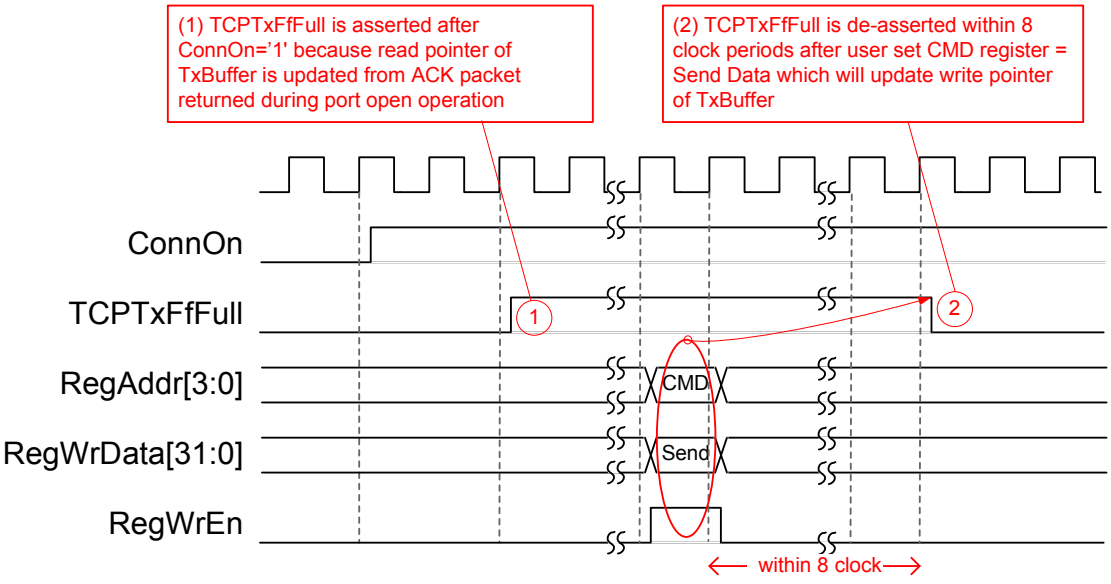

Figure 6: TCPTxFfFull de-asserted after open connection

Rx FIFO Interface

When IP core receives data from external, data will be stored in Rx Data buffer. User can read data from this buffer through FIFO interface, as shown in Figure 7. User can monitor data available status from TCPRxFfEmpty. Data can be read by setting TCPRxFfRdEn='1' when TCPRxFfEmpty is cleared to '0'. TCPRxFfRdData will be valid in next clock period. TCPRRxFfRdEn must be de-asserted to'0' at the same clock with TCPRxFfEmpty = '1'. All data in Rx data buffer will be flushed from open connection detect, which can be monitored from TCPRxFfFlush signal.

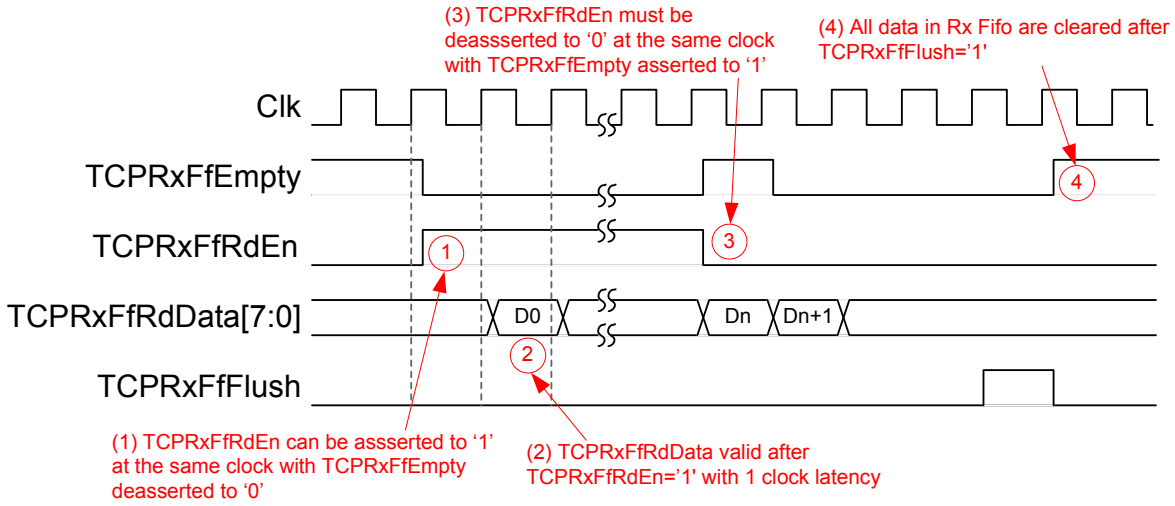

Figure 7: Rx Data Buffer Interface by Empty flag Timing Diagram

Rx data buffer status can be also monitored by using TCPRxFfRdCnt to design burst read transfer. This signal shows total number of available data in Rx data buffer. So, user can assert TCPRxFfRdEn='1' for many clock periods which is not more than the value of TCPRxFfRdCnt, as shown in Figure 8.

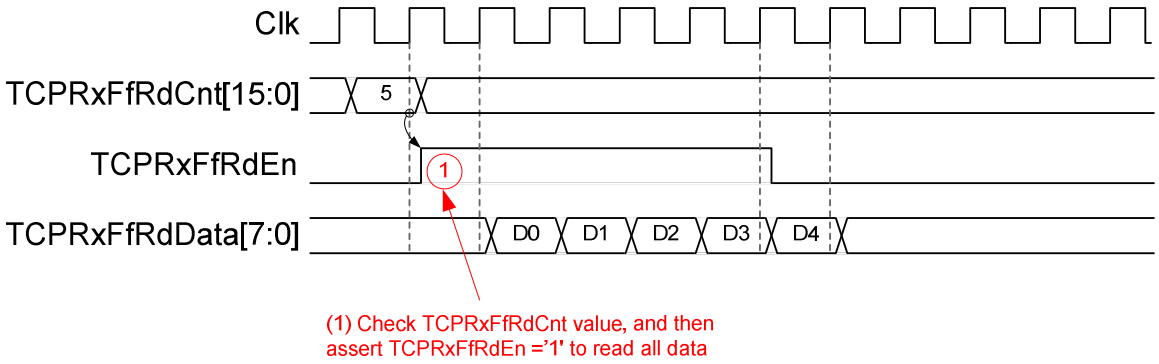

Figure 8: Rx Data Buffer Interface by Read counter Timing Diagram

EMAC Interface

For EMAC interface, timing diagram is compatible to Xilinx TEMAC IP core. As shown in Figure 9, to transmit packet TOE1G-IP will assert MacTxValid with the first data of the packet. Both signals will be latched until MacTxReady output from EMAC is asserted to '1' to acknowledge data transmit request. MacTxReady must be asserted to '1' until the packet is end of transmission. MacTxLast and MacTxValid will be asserted to '1' with the last transmit data to show end-of-packet status.

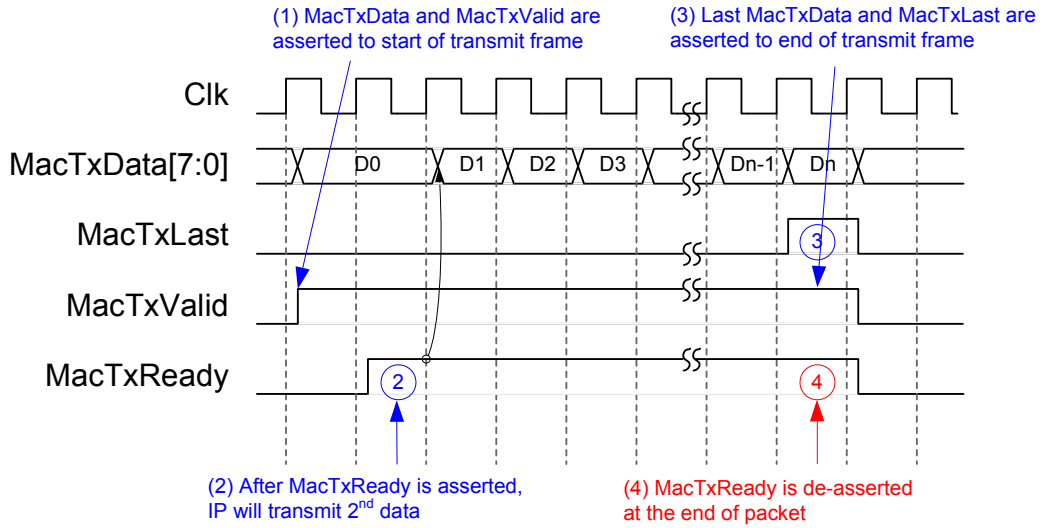

Figure 9: Transmit EMAC Interface

Figure 10 shows timing diagram of Received side. TOE1G-IP monitors start of received frame from MacRxValid which changes from '0' to '1'. MacRxData will be received continuously until MacRxLast is asserted to be end of packet. Last MacRxData (D_n) must be available on the bus after D_{n-1} because TOE1G-IP will read D_0-D_n continuously. MacRxLast is used to be valid signal of MacRxUser.

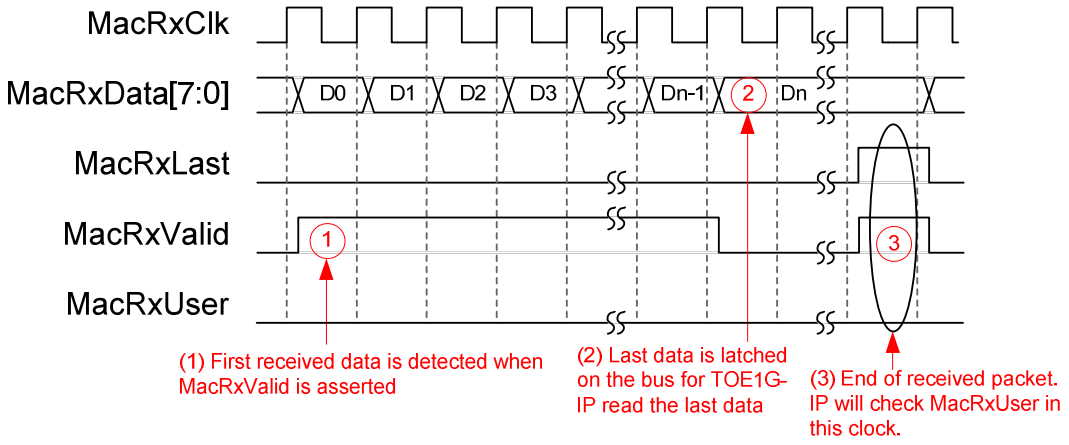

Figure 10: Received EMAC Interface

Verification Methods

The TOE1G-IP Core functionality was verified by simulation and also proved on real board design by using AC701/KC705/VC707/ZC706/Zynq Mini-ITX evaluation board.

Recommended Design Experience

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

Ordering Information

This product is available directly from Design Gateway Co., Ltd. Please contact Design 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	Dec-03-2012	New release
1.1	Dec-11-2012	Update IP port and buffer size description
1.2	Nov-19-2013	Update Figure1
1.3	Nov-28-2013	Add AC701 Support
1.4	Mar-13-2014	Update port name and add more description for Transmit operation
1.5	Apr-24-2014	Add VC707 Support
2.0	Aug-07-2014	Update IP to support full-duplex
2.1	Nov-20-2014	Add TCPTxFfFull condition
2.2	Jan-19-2015	Support Zynq device and add PSH register
2.3	Dec-24-2015	Add Busy status and register readback
2.4	Sep-2-2016	IP core renamed from TOE2-IP to TOE1G-IP
2.5	Nov-14-2016	Add Zynq Mini-ITX support and add WIN register
2.6	May-11-2017	Add EMACIF block to support new version of Xilinx EMAC