TOE1G-IP reference design manual

Rev1.4 14-Nov-16

1. TCP/IP

TCP/IP is the core protocols of the Internet Protocol Suite for networking application. TCP/IP model has four layers, i.e. Application Layer, Transport Layer, Internet Layer, and Network Access Layer. In Figure 1, five layers are displayed for simple matching with hardware implementation by FPGA. Network Access Layer is split into Link and Physical Layer.

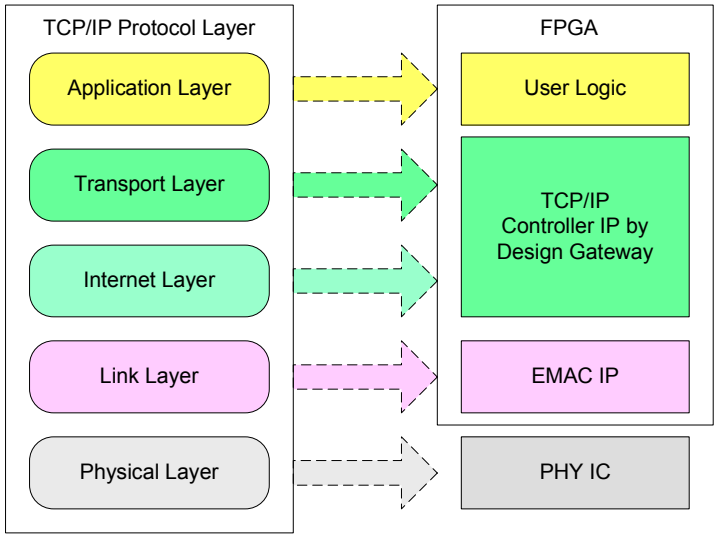

Figure 1 TCP/IP Protocol Layer

TOE1G-IP implements Transport and Internet layer of TCP/IP Protocol. For transmit side, TOE1G-IP will split TCP data from user logic into packet format with TCP and IP header generation and then send out to EMAC. For received side, TOE1G-IP will extract TCP data and header from IP packet and then store only TCP data in buffer for user logic reading.

By using TCP, data reliability is guaranteed by monitoring acknowledge packet. For transmit side, TOE1G-IP will retransmit data if acknowledge packet is lost. For received side, if TOE1G-IP detects that some received data is lost, it will send duplicate acknowledge packet to request the lost packet.

The lower layer protocols are implemented by EMAC-IP from Xilinx and external PHY chip.

This reference design provides evaluation system which includes simple user logic to send and receive data with TOE1G-IP. This system demonstrates on FPGA Development board to operate with Test application on PC for transferring high speed data on network. More details are described as follows.

2. Hardware description

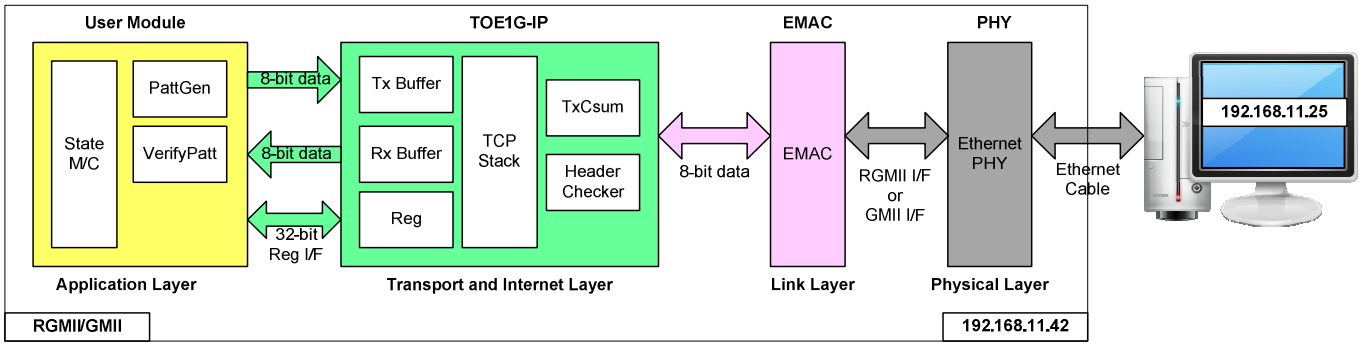

Figure 2 Hardware Architecture in FPGA board by RGMII or GMII (AC701/KC705)

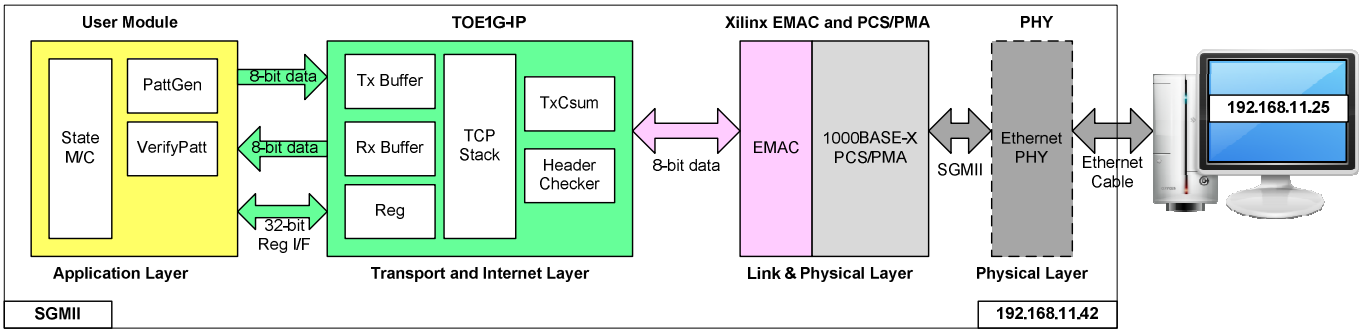

Figure 3 Hardware Architecture in FPGA board by SGMII (VC707/Zynq Mini-ITX)

As shown in Figure 2 and Figure 3, hardware architecture can be divided into 4 modules to support each TCP/IP layer protocol. TOE1G-IP operates with EMAC and external PHY to implement all four lower layers of TCP/IP Protocol. User prepares modules to transfer TCP data and write/read control signal with TOE1G-IP for data transferring with test application on PC. This reference design includes the example of User Module to generate Test pattern for transmit and verify Test pattern for received side based on FPGA development board.

- External PHY

Typically, physical layer is implemented by external PHY chip by using GMII, RGMII, or SGMII. For SGMII mode, it can set to use internal transceiver within FPGA for connecting 1G Ethernet with external network instead of using external PHY.

- EMAC

Link layer is implemented by EMAC-IP (Tri Mode Ethernet MAC with 1000 Mbps speed), provided by Xilinx.
https://www.xilinx.com/products/intellectual-property/temac.html

The setting of EMAC-IP is followed.

- PHY Interface is RGMII/GMII/SGMII
- MAC speed is fixed to 1000 Mbps
- Configuration is Ver
- MDIO I/F is disable

Data interface of EMAC can connect with TOE1G-IP directly.

- TOE1G-IP

More details about both modules are described in “dg_toe1gip_data_sheet_xilinx_en.pdf” document.
http://www.dgway.com/products/IP/TOE1G-IP/dg_toe1gip_data_sheet_xilinx_en.pdf

- User Module

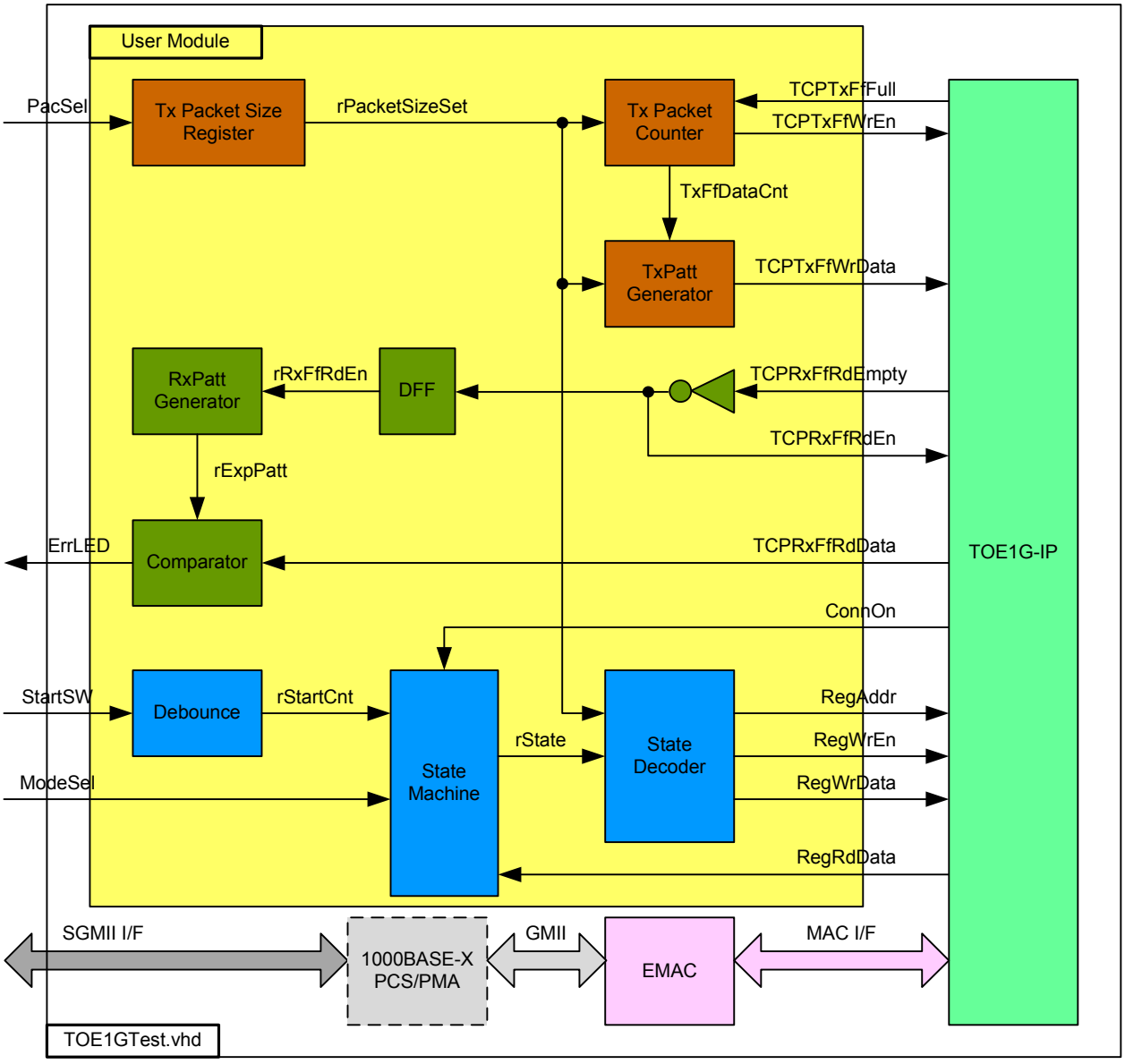

Figure 4 User Module and Test System block diagram

User Module can split into three parts, i.e. TxFIFO interface, RxFIFO interface, and Register Control interface. 32-bit increment test data is increased by 1 every end of each Tx packet for Transmit side. Tx Packet counter is designed to count the number of data in each Tx packet which can be set by external PacSel DIPSW. There are two supported sizes in this demo, i.e. 1460 bytes for non-Jumbo frame, and 8960 bytes for Jumbo frame.

32-bit increment data is also generated by RxPatt Generator to verify data output from RxFIFO interface of TOE1G-IP. Simple logic to read out data from FIFO by monitoring Empty flag is designed. ErrLED is Blink when data verification is error.

Register Control interface is designed by using State Machine. Register address and write value signals are decoded from State Machine. Transfer data direction is selected by ModeSel DIPSW, and data starts transferring when StartSW is pressed by user. State Machine diagram is displayed as shown in Figure 5.

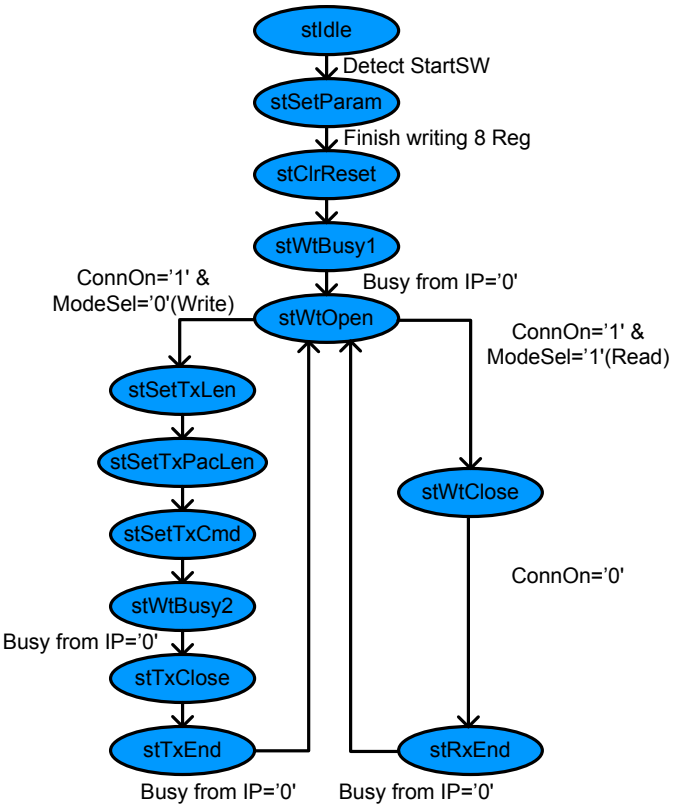

Figure 5 State Machine Diagram within User Module

State machine changes to stSetParam state after user press StartSW button. In stSetParam state, it will set parameters to register within TOE1G-IP, i.e.

- Source MAC address (SML/SMH Reg) = 00:01:02:03:04:05
- Source IP address (SIP Reg) = 192.168.11.42
- Source Port number (SPN Reg) = 4000
- Destination IP address (DIP Reg) = 192.168.11.25

Next State, stClrReset, is applied to release Reset signal (RST Reg=0) within TOE1G-IP and then starting parameter initialization. State machine waits initialization complete by monitoring Busy flag (bit0 of CMD Reg) through Register interface. Then, state will pause in stWtOpen until test application on PC starts running.

In this demo, FPGA runs as Server mode and Test application on PC runs as Client mode, so the connection is opened by Test application on PC. At FPGA side, ConnOn output from TOE1G-IP will change to '1' when Client is connected, and then state machine will change to stSetTxLen for transmit mode or to stWtClose for received mode.

On transmit mode, more three states, i.e. stSetTxLen, stSetTxPacLen, and stSetTxCmd are designed for setting total transfer size (TDL Reg), packet size (PKL Reg), and data sending to command register (CMD Reg=0) within TOE1G-IP. Then, Busy signal is monitored to wait transfer complete in stWtBusy2 state. After all data are transferred completely, State sends command to TOE1G-IP for sending packet to close connection (CMD Reg=0x3) in stTxClose state. After connection is closed and Busy = '0', state will run in stWtOpen for waiting next transfer.

On received mode, state will stay in stWtClose to wait data transfer from PC complete and follow with connection closed command from PC. So, ConnOn value from TOE1G-IP will change from '1' to '0' after connection has already closed. Similar to transmit mode, state will go back to stWtOpen for waiting next transfer.

In conclusion, the demo uses passive open mode for both transmit and received transfer, but uses both modes to close connection. Active close connection is operated for data transmit mode while passive close connection is operated for data received mode.

3. Test Software description

Two test applications are applied within this demo, i.e. “recv_tcp_client” and “send_tcp_client”. Both application runs as client mode.

- recv_tcp_client

This test application runs to test sending operation of TOE1G-IP, so data sending to PC will be verified by test application. This application requires three input parameters from user, i.e.

- FPGA IP address: This demo sets IP address to “192.168.11.42”. User can modify HDL code of User module to change this value.
- FPGA Port number: This demo sets Port number to “4000”. User can modify HDL code of User module to change this value.
- Packet size: This demo can set as two values, i.e. 1460 for non-Jumbo frame mode, and 8960 for Jumbo frame mode. If setting with wrong value, verified error message will be displayed on Test application and operation will be stopped.

The operation sequence of the application is follows.

(1) Get three parameters from user.
(2) Create socket and then set properties of received buffer.
(3) Set IP address and Port number from user parameter and then connect.
(4) Loop run for receiving data and data verification. Data format is 32-bit increment value which increases every end of packet. Thus, all data in same packet will be similar. Two errors can be printed out from verification process, i.e. “Drop Expect” printed out when 1st data of packet is not expect value, and “Error Expect” printed out when data within each packet is not expect value. Received data size is printed out on console every second.
(5) Socket is closed by FPGA side and then Performance with total number of transferred data will be printed out as test result.
(6) Go back to step (3) in loop to continue test operation until operation cancel.

- send_tcp_client

This test application sends data out to TOE1G-IP to test received operation. This application requires four input parameters from user, i.e.

- FPGA IP address: This demo sets IP address to “192.168.11.42”. User can modify HDL code of User module to change this value.
- FPGA Port number: This demo sets Port number to “4000”. User can modify HDL code of User module to change this value.
- Packet Count: This value is set transfer size in 16kByte unit. Total transfer size is equal to this value x 16kByte. Valid range is 1-262143.
- Verification On/Off: Select ‘0’ to transfer dummy data and ‘1’ to transfer 32-bit increment data out. This setting value is effect to output performance from PC. In some PC, performance in dummy data mode is better than increment data mode.

The operation sequence of the application is follows.

(1) Get three parameters from user.
(2) Create socket and then set properties of transmit buffer.
(3) Set IP address and Port number from user parameter and then connect.
(4) Fill test pattern with dummy (all ‘0’) or increment pattern to buffer and then send data out. Transfer size is set from user.
(5) Close socket and print out performance with total number of transferred data as test result.

4. Revision History

Revision	Date	Description
1.0	7-Aug-14	Initial Release
1.1	29-Dec-14	Add ZC706 support
1.2	29-Dec-15	Update recv_tcp_client message
1.3	2-Sep-16	IP core product renamed from TOE2-IP to TOE1G-IP
1.4	14-Nov-16	Update Figure2- Figure3