UDP1G-IP reference design manual

Rev1.0 2-Mar-17

1. Introduction

Comparing to TCP, UDP provides a procedure to send messages with a minimum of protocol mechanism, but the data cannot guarantee because of no handshaking dialogues. Like TCP, UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram.

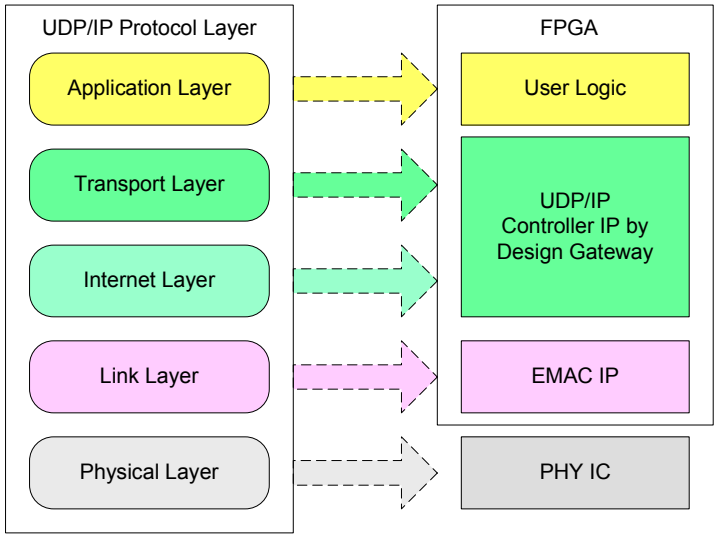

Figure 1 UDP/IP Protocol Layer

UDP1G-IP implements Transport and Internet layer of UDP/IP Protocol. For transmit side, UDP1G-IP will prepare UDP data from user logic and add UDP/IP header to generate Ethernet packet format before sending out to EMAC. For receive side, UDP1G-IP will extract UDP data from Ethernet packet. UDP/IP header will be verified to check packet valid. If packet is valid, UDP data will be extracted and stored in buffer for user logic reading.

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

This reference design provides evaluation system which includes simple user logic to send and receive data by using UDP1G-IP. This system demonstrates on Intel Development board to operate with Test application on PC for transferring high speed data on network. More details are described as follows.

2. Environment

To operate this reference design, following environment must be setup.

- FPGA Development board
- QuartusII Programmer
- Ethernet cable (Cat5e or Cat6)
- PC with Gigabit Ethernet
- USB A-B/Micro-B cable for FPGA configuration
- Test Application, i.e. “send_udp_client” and “recv_udp_client”, provided by Design Gateway

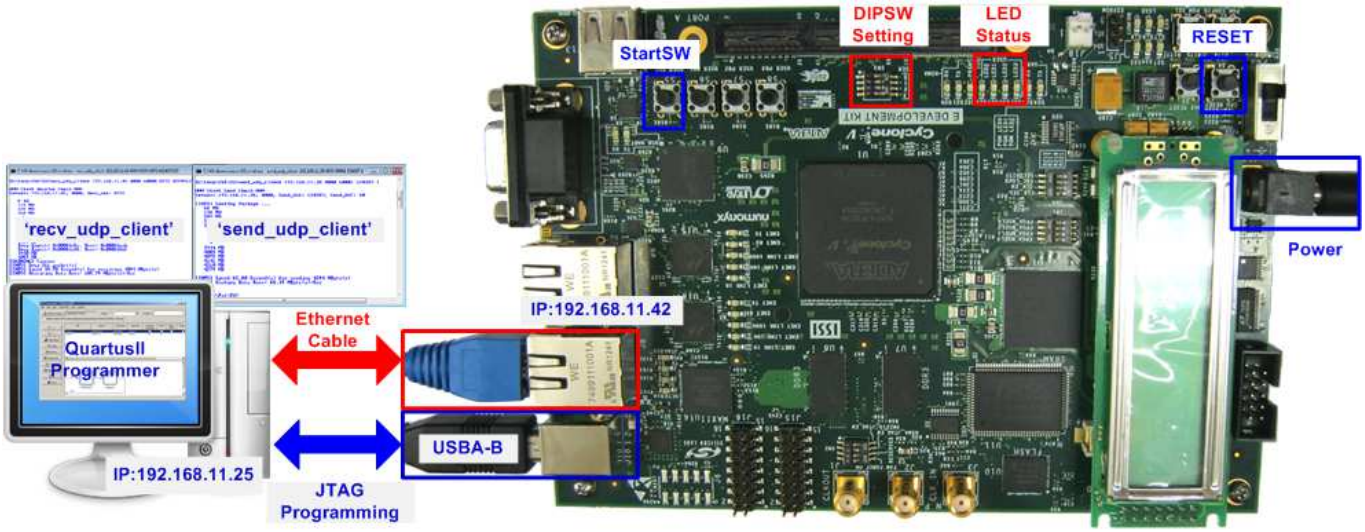

Figure 2 UDP1GIP Demo on CycloneV E board

3. Hardware description

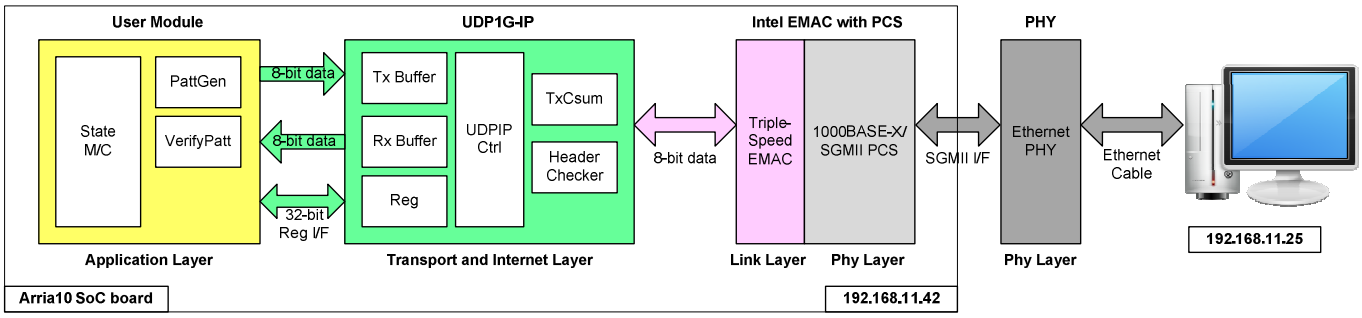

Figure 3 Hardware Architecture in CycloneV E/ArriaV GX reference design (RGMII)

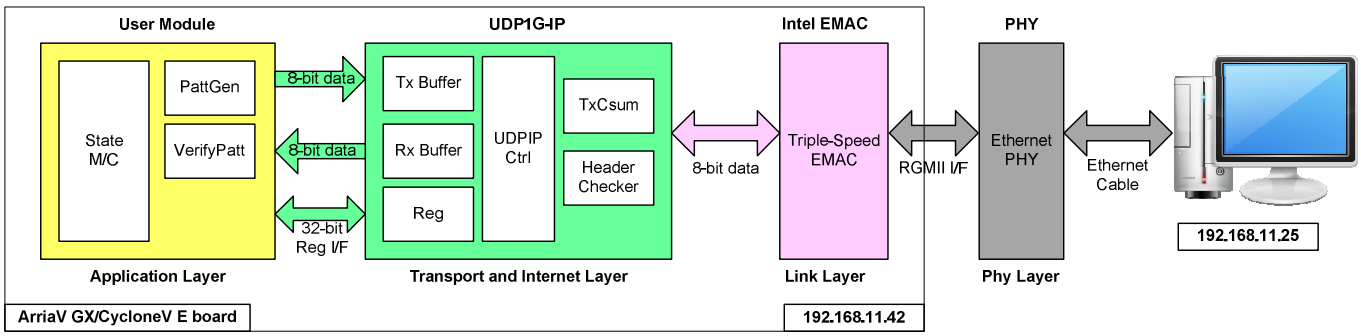

Figure 4 Hardware Architecture in Arria10 SoC reference design (SGMII)

As shown in Figure 3, hardware architecture can be divided into 4 modules to support each UDP/IP layer protocol. UDP1G-IP operates with EMAC and external PHY to implement all four lower layers of UDP/IP Protocol. The reference design can transfer data in both directions by running UDP1G-IP with two test applications on PC at the same time.

For FPGA to PC direction, UDP data is generated by test pattern generator inside User module. Test data will be verified by test application on PC (recv_udp_client.exe). For PC to FPGA direction, data generated by test application on PC (send_udp_client.exe) will be verified by User module. State machine is designed to set and monitor UDP1G-IP command and status through 32-bit register interface. Data interface between UDP1G-IP and User Module is 8-bit FIFO interface for both directions.

- External PHY

Physical layer is implemented by external PHY chip. The interface type of PHY chip can be three formats, i.e. SGMII (Arria10 SoC board), RGMII (ArriaV GX Starter board/CycloneV E board), or GMII.

- EMAC

Link layer and PCS/PMA are implemented by Triple-speed Ethernet MAC, provided by Intel. EMAC has two user interfaces, i.e. Avalon stream for transferring data and Avalon-MM for configuration. In demo system, Avalon stream of EMAC is connected to UDP1G-IP while Avalon-MM interface is connected to EMACCtrl module to configure EMAC. EMACTrl includes state machine which runs only one time to initialize basic parameters of EMAC and external PHY. The details about the register in EMAC are described in “Configuration Register Space” topic within “Triple-Speed Ethernet MegaCore Function User Guide” document, provided by Intel.
https://www.altera.com/en_US/pdfs/literature/ug/ug_ethernet.pdf

The sequence of EMAC State machine for SGMII and RGMII are different. For SGMII mode, EMAC state machine will program only Base Configuration area to configure MAC function such as disable/enable transmit and receive paths, frame length, Tx IPG length, and software reset.

For RGMII mode, EMAC state machine will program both Base Configuration and MDIO Space1 area. MDIO Space1 area is used to access external PHY register through MDIO interface for enable RGMII Receive/Transmit timing control function. When enable timing control bit, transmit and receive clock will include the delay to shift clock phase for synchronous with transmit/receive data.

- UDP1G-IP

More details about both modules are described in “dg_udp1gip_data_sheet_intel_en.pdf” document.

- User Module

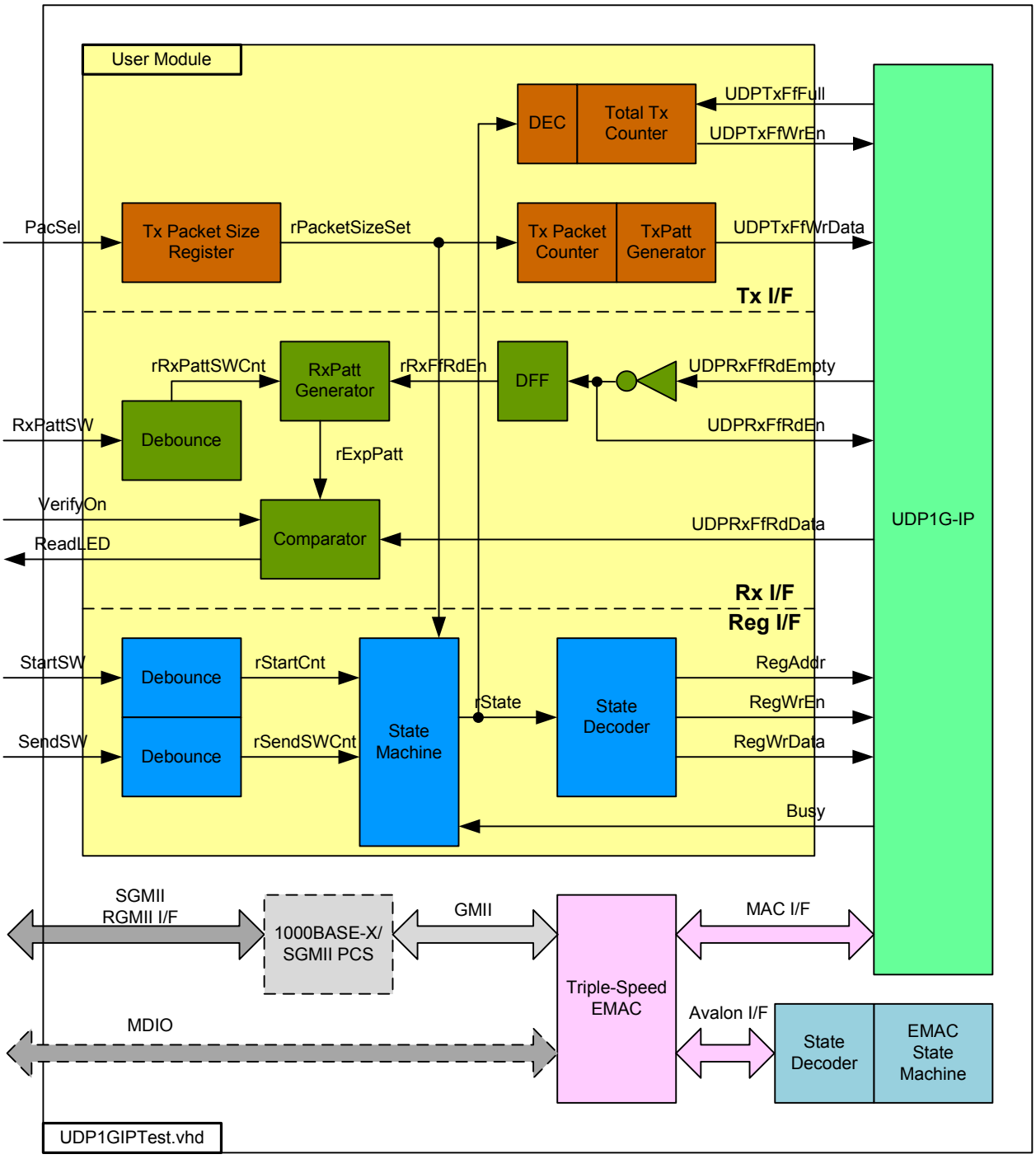

Figure 5 User Module and Test System block diagram

User Module can split into three parts, i.e. TxFIFO interface, RxFIFO interface, and Control interface.

Tx Interface

For transmit operation, TxPatt Generator generates 32-bit increment test data to be output for TxFIFO. Pattern is increased every end of each Tx packet, so Tx Packet Counter is designed to count the number of data in each Tx packet. Two packet sizes can be selected from PacSel DIPSW, i.e. 1472 for non-jumbo frame size, or 8972 for jumbo frame size. Test pattern will be generated during transmit operation until equal to total transfer size. State machine is decoded to confirm that user requests to run new transmit test. Total Tx Counter is used to count total numbers of test data, which is fixed to 0xFFFF_FFFF (4GB) in the reference design.

Rx Interface

32-bit increment data is also generated by RxPatt Generator to verify data output from RxFIFO interface of UDP1G-IP. The logic to read out data from FIFO is simply designed by monitoring Empty flag. ReadLED is blink when read data is not equal to expected data from RxPatt Generator and data verification is enabled. RxPattSW is used to reset Start value of test pattern.

Reg Interface

Control interface is designed by using State Machine. Register address and write value signals are decoded from State Machine to program parameter for initialization process and set packet size/transfer size/command register for transmit operation. Transmit operation will start after SendSW is pressed by user. State Machine diagram is shown in Figure 6. Busy flag output from IP is used to monitor status that IP operation has been completed.

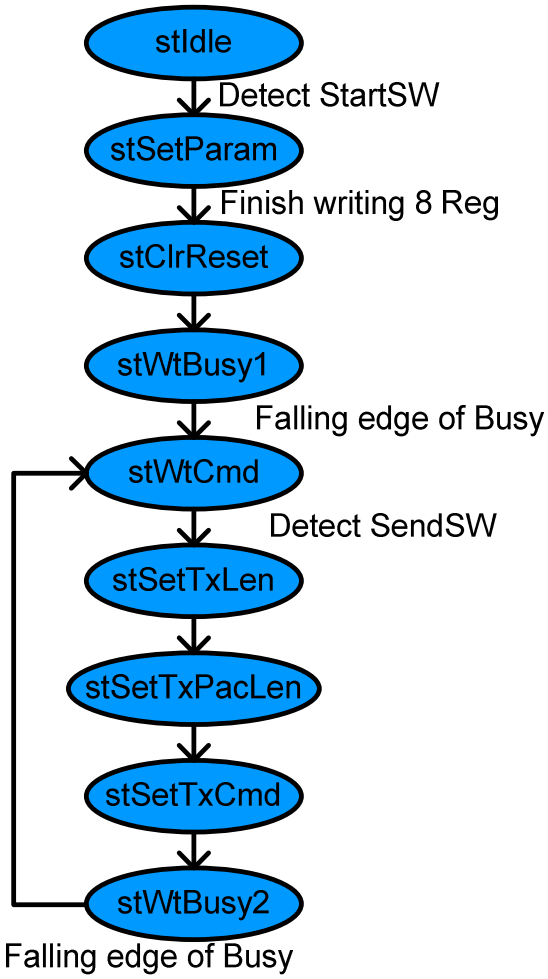

Figure 6 State Machine Diagram within User Module

State machine changes to stSetParam state after user press StartSW button. In stSetParam state, it will set parameters to register within UDP1G-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
- Destination Port number for Tx(DPN Reg[15:0]) = 60000
- Destination Port number for Rx(DPN Reg[31:16]) = 60001

Next State is stClrReset which is applied to release Reset signal (RST Reg=0) within UDP1G-IP and then IP starts parameter initialization. State machine waits initialization complete by monitoring falling edge of Busy signal. Then, state will stay in stWtCmd which is Idle state to wait command from user.

For data transmit operation, state machine is designed to set command to send 4GB test data from FPGA to PC. After user press SendSW button, state machine will go to stSetTxLen for setting total transfer size (TDL Reg), stSetTxPacLen for setting packet size (PKL Reg), and stSetTxCmd for setting command register (CMD Reg). Then, Busy signal is monitored to wait transfer complete in stWtBusy2 state. After all data are transferred completely, it will go back to stWtCmd for waiting next command.

For receive mode, IP can transfer UDP data from PC to user logic without setting any register by State machine. So, user can test data transfer in both directions at the same time.

4. Test Software description

Two test applications are applied within this demo, i.e. “recv_udp_client” and “send_udp_client”. To run both applications at the same time, port number at PC side will be set by different value. 60000 is used for FPGA to PC direction, while 60001 is used for PC to FPGA direction.

- recv_udp_client

This test application runs to test sending operation of UDP1G-IP, so data sending to PC will be verified by test application. This application requires five input parameters from user which must be the same value setting in HDL code of UserModule. User can change the value by modifying constant in HDL code. The default value in reference design is follows.

- Dst_Addr: IP address of FPGA. Set to “192.168.11.42” for this demo.
- Dst_Port: FPGA Port number. Set to “4000” for this demo.
- Src_Port: PC Port number. Set to “60000” for this demo.
- Recv_Len: Packet size in byte unit. Set to “1472” for non-Jumbo frame mode or “8972” for Jumbo frame mode. If setting with wrong value, verified error message will be displayed on Test application and operation will be stopped.
- Total_Len: Total transfer size in byte unit. Set to “4294967295” for this demo.

The operation sequence of the application is follows.

(1) Get parameters from user.
(2) Create socket and then set properties of receive buffer.
(3) Set IP address and Port number from user parameter and then connect.
(4) Loop to verify data until total receive data is equal to set value or no more receive data within 0.5 sec. Verification pattern is 32-bit increment starting from 0. Pattern is increased after end of each packet size (1472 or 8972 byte). So, all data in one packet will be similar. Two messages can be printed out from verification process, i.e.
   - o “Drop Expect” when 1st data of packet is not equal to expect value. This is warning message and application still continue to verify data.
   - o “Error Expect” when data within the packet is not equal to 1st data. This case is error condition, so application will stop operation.

   During running, application will print total receive size on the console every second.
(5) a) Socket is closed by PC. Application will show performance with total number of receive data and total drop packet to be test result.
   b) UDP transfer sometimes losses data packet, so test application includes 0.5 sec timeout operation. If there is no receive data for 0.5 sec, test application will exit data transfer loop and close the socket. In this case, “Timeout” message will be displayed to show user that the application receives data less than the setting value.

- send_udp_client

This test application sends data out to UDP1G-IP to test receiving operation. Similar to recv_udp_client, this application requires five input parameters from user, which is fixed value from HDL code of UserModule.

- Dst_Addr: IP address of FPGA. Set to “192.168.11.42” for this demo.
- Dst_Port: FPGA Port number. Set to “4000” for this demo.
- Src_Port: PC Port number. Set to “60001” for this demo. Different port number from recv_udp_client is used to allow user run two test applications at the same time.
- Packet Count: Total number of 8kByte packet to send out to FPGA. Total byte size is equal to this value x 8096 byte. Valid range is 1-524287.
- Verification On/Off: Set ‘0’ to transfer dummy data or ‘1’ to transfer 32-bit increment data. When running in dummy mode, PC resource will be less than increment data. So, in some PC environment performance of dummy mode will be better than increment data mode.

The operation sequence of the application is follows.

(1) Get 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. Each packet size is fixed to 8096 byte unit. By using this packet size, PC which can support jumbo-frame will generate UDP packet without fragmentation.
Note: IP does not support to receive IP packet which has fragment.
(5) Close socket and print out performance with total number of send data as test result.

5. Revision History

Revision	Date	Description
1.0	2-Mar-17	Initial Release