Design And VHDL Implementation Of HDLC Controller on FPGA Sparten 6

Saurabh Pareek[1], Mr. Rakesh Kumar[2]
[1, 2] Dept of VLSI Design
[2]Assistant Professor, Dept of ECE,
[1, 2] Sobhasaria Group of Institutions, Sikar (Raj.)

Abstract- *For communication of data over a network a protocol is required. The layer-2 of OSI model defines such protocols. HDLC is one of these protocols. HDLC is one of the most enduring and fundamental standards in Communications. HDLC is in itself a group of several protocols or rules for transmitting data between network points. The HDLC protocol also manages the flow or pace at which the data is sent. The data is organized into a unit called a frame. HDLC controllers are devices, which executes the HDLC protocol. Some of the key operations of the HDLC protocol implemented are handling bit-oriented protocol structure and formatting data as per the packet switching protocol, it includes Transmitting and receiving the packet data serially and providing the data transparency through zero insertion and deletion. This dissertation discusses the design and implementation of some such HDLC controllers.*
HDLC controller is a high performance module for the bit oriented, switched, non-switched packet transmission module. It supports half duplex and full duplex communication lines, point-to-point and multipoint channels. Furthermore, the controller is designed to permit synchronous, code transparent data transmission. The data stream and transmission rate is controlled by the network node. In this dissertation, we have designed, simulated and implemented HDLC controller. This design is coded in a hardware description language (VHDL). The function of coded design is to simulate on simulation software (e.g. ISIM Simulator). After simulation, the design is synthesized and translated into a structural architecture, in terms of the components on the target FPGA device (Spartan 6) and perform the post-translate simulation in order to ensure the proper functioning of the design after translation. After the successful simulation of the post-translate model, the design is mapped to the existing slices of the FPGA and the post-map model simulated. The post-map model does not include the routing delays. In this dissertation, we implemented the various HDLC controllers for crc-16 of 16-bit address and 8-bit data, for 16-bit address and 16-bit data, for 32-bit address and 32-bit data, for crc-32 of 16-bit address and 8-bit data and for 16-bit address and 8-bit data and 32-bit crc, simulation result for final output at the receiver end for 8- bit data16-bit address and 16-bit CRC, with bit stuffing and removal of error in HDLC.

I. INTRODUCTION

HDLC [High-level Data Link Control] is a group of protocols for transmitting [synchronous] data [Packets] between [Point-to-Point] nodes. In HDLC, data is organized into a frame. HDLC protocol resides with Layer 2 of the OSI model, the data link layer. It is an efficient layer 2 protocol standardized by ISO for point-to-point and multipoint data links. HDLC provides minimal overhead to ensure flow control, error control, detection and recovery for serial transmission.

The HDLC frame is synchronous and therefore relies on the physical layer to provide method of clocking and synchronizing the transmission and reception of frames. The frames are separated by HDLC flag sequences that are transmitted between each frame and whenever there is no data to be transmitted. To inform the receiving station that a new packet is arriving and synchronizes the receive clock with the transmitted clock a specific bit pattern is added at the front and the back of the packet. The header of the packet contains an HDLC address and an HDLC control field. The specific bit pattern is used to affix with the packet in the case of HDLC Controller is 01111110. The length of the address field is normally 0, 8 or 16-bits in length. In many cases the address field is typically just a single byte, but an Extended Address [EA] bit may be used allowing for multi-byte addresses. A one residing in the LSB bit indicates [the end of the field] that the length of the address field will be 8-bits long. A zero in this bit location [now the first byte of a multi-byte field] indicates the continuation of the field [adding 8 additional bits]. The Control field is 8 or 16-bits and defines the frame type; Control or Data To guarantee that a flag does not appear inadvertently anywhere else in the frame, HDLC uses a process called bit stuffing. Every time the user wants to send a bit sequence having more than 5 consecutive 1s, it inserts (stuffs) one redundant 0after the fifth 1. The trailer is found at the end of the frame, and contains a Cyclic Redundancy Check (CRC), which detects any errors that may occur during

transmission. A CRC value is generated by a calculation that is performed at the source device. The destination device compares this value to its own calculation to determine whether errors occurred during transmission. First, the source device performs a predetermined set of calculations over the contents of the packet to be sent. Then, the source places the calculated value in the packet and sends the packet to the destination. The destination performs the same predetermined set of calculations over the contents of the packet and then compares its computed value with that contained in the packet. If the values are equal, the packet is considered valid. If the values are unequal, the packet contains errors and is discarded. The receiver can be configured into transparent mode, effectively disabling the HDLC protocol functions. In normal HDLC protocol made, all received frames are presented to the host on the output register.

Features

The main features are:

1. Full duplex and half duplex modes of operation
2. Automatic frame check sequence generation and checking.
3. Minimum CPU overhead.
4. CCITT X.25 compatible.
5. Capable of working in various modes.
6. Single +5V Supply

History of Internetworking

The layered concept of networking was developed to accommodate changes in technology. Each layer of a specific network model may be responsible for a different function of the network. Each layer will pass information up and down to the next subsequent layer as data is processed. [1]

OSI 7 Layers Reference Model for Network Communication

Open Systems Interconnection (OSI) model is a reference model developed by ISO (International Organization for Standardization) in 1984, as a conceptual framework of standards for communication in the network across different equipment and applications by different vendors. It is now considered the primary architectural model for inter-computing and internetworking communications. Most of the network communication protocols used today have a structure based on the OSI model. The OSI model defines the communications process into 7 layers, which divides the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layers. Each layer is reasonably self-contained so that the tasks assigned to each layer can be implemented independently. This enables the solutions offered by one layer to be updated without adversely affecting the other layers. [1]

II. AIMS AND OBJECTIVE

The objective of thesis is to design and implement a high performance HDLC Controller for the bit oriented, switched, and non switched, packet transmission to permit synchronous, code transparent data transmission. Initially, the digital design (block diagram) will be drafted showing the basic functioning of the hardware in terms of the blocks. This will then be coded in a VHSIC Hardware Description Language (VHDL). The functioning of the coded design is to be simulated on simulation software (e.g. ModelSim). After proper simulation, the design is to be synthesized and then translated to a structural architecture in terms of the components on the target FPGA device (Spartan 3) and the perform the post-translate simulation in order to ensure the proper functioning of the design after translation. After the successful simulation of the post-translate model the design is mapped to the existing slices of the FPGA and the post-map model simulated. The post-map model doesn't include the routing delays. After the successful completion of the post-map simulation, the design is then routed and a post-route simulation model with the appropriate routing delays is generated to be simulated on the HDL simulator. After this a programming file is generated to program the FPGA device. The objective is to run the programmed FPGA at a frequency as high as possible.

III. LITERATURE SURVEY

The CDAC HDLC controller operates at the data link layer of the OSI Model. Hence the main focus of the survey is to understand the data link layer and develop a protocol which can offer its services to the layer above it i.e. is the network layer and the layer below it i.e. the physical layer.

The main function of this protocol controller is to perform a number of separate activities like physical addressing, to check for errors, flow control etc.

The research papers on the design of HDLC protocols are published in various journals and presented in many conferences. Here the paper selected describes the design of protocol using VHDL or Verilog language which includes the design of transmitter and receiver. Some of the papers present the design of protocol for particular application. They have designed the protocol according to the

requirements of application. The clock requirement; synchronization requirement for every design is different. The CRC polynomial selected can be 16-bit or 32-bit. Mostly CRC-16 has been used.Most papers have used FSMs to design the transmitter and receiver of their controller.

N. Mrudula K. Babulu (2016), The data link layer is the second layer of the seven layered Open Systems Interface (OSI) model. It responds to service requests from the network layer and issues service requests to the physical layer. The data link layer transforms the physical layer which is a raw transmission facility, to a reliable link. It makes the physical layer appear error free to the network layer.

IV. XILINX ISE DESIGN SUITE

History

In this modern era, **Xilinx ISE** (**I**ntegrated **S**yndissertation **E**nvironment) is a very imperative software tool created by **Xilinx** which provides various functions such as for blend and scrutiny of HDL designs, helps in permitting the designer to synthesize their designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to diverse stimuli and also establish the objective method with the programmer. **Xilinx** is a technology developed by an American company, mainly a purveyor of programmable logic devices. This technology is also known for originating the field programmable gate array (FPGA) and be the first semiconductor company with a fables built-up model.
In 1984, Silicon Valley is one of the company which is headquartered in San Jose, California to establish this technology, with additional offices in Longmont, Colorado; Dublin, Ireland; Singapore; Hyderabad, India; Beijing, China; Shanghai, China; Brisbane, Australia and Tokyo, Japan. The one of the foremost FPGA products.

families include Virtex (high-performance), Kintex (mid-range) and Artix (low-cost), and the retired Spartan (low-cost) series and it also includes chief computer software such as Xilinx ISE and Vivado Design Suite [30][33].

V. SIMULATION RESULTS

After the design and implementation of the HDLC Controller, the results obtained are as follows:

Simulation result for 8-bit data, 8-bit address and 16-bit CRC Check

Simulation result for 8-bit data, 8-bit address and crc-16

For the data<=00001110and8-bit address<=11110000, we make clock=1, reset=0, wrtaddresshi=0 and wrtaddresslo=1. After the address and the data are attached together, we divide them with a constant polynomial and append the remainder of the division along the data and address. The simulation result for the generation of crc1 is highlighted below-

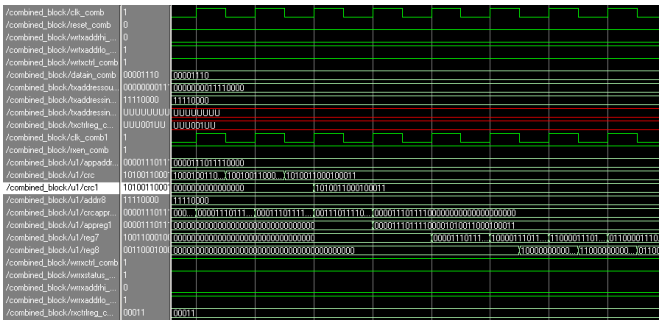

Fig. 5.1: Simulation result for crc-16 of 16-bit address and 8-bit data

5.1.2 Simulation result of the final O/P at the receiver end for 8-bit and 16-bit crc

At the receiver the data input, address and the appendedcrc is again divided with the same constant polynomial and if the crc2 comes out be zero, it shows an error free reception of the packet. The receiver O/P i.e. rxdataout<=00001110 and rxaddressout<=0000000011110000 which is same as that of transmitter.

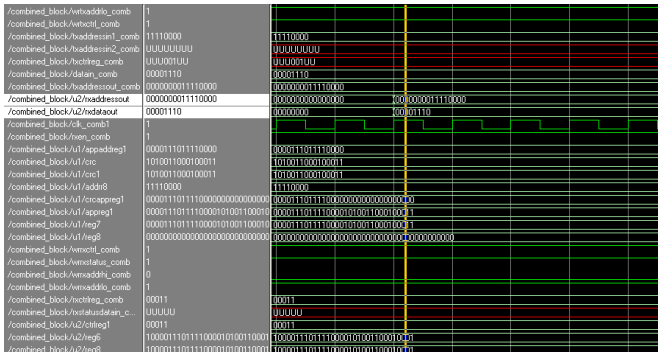

Fig. 5.2: Simulation result for O/P at the receiver for 16-bit address and 8-bit data

Simulation result for 16-bit address, 8-bit data and 16-bit CRC Check

Simulation result for 16-bit address, 8-bit data and crc-16

For the data<=00110011 and 16bitaddress i.e. txaddressinlo<=11110000, txaddressinhi<=11110000 we make clock=1, reset=0, wrtaddresshi=1 and wrt addresslo=1.

After the address and the data are attached together, we divide them with a constant polynomial of 16-bits and append the remainder of the division along the data and address. The simulation result for the generation of crc1 is highlighted below:

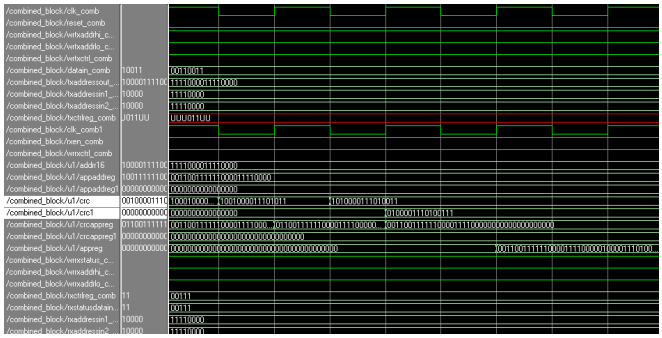

Fig. 3: Simulation result for crc-16 of 16-bit address and 8-bit data

Simulation result of the final O/P at the receiver end for 8-bit data, 16bit address and 16-bit crc

The highlighted simulation results shown below shows the result of the CRC check at the O/P which is crc2<=000000000000000 which indicates an error free transmission. The resultant address, data at the O/P which is same as that of transmitter is. Rxdatout<=00001110 and rxaddressout<=1111000011110000.

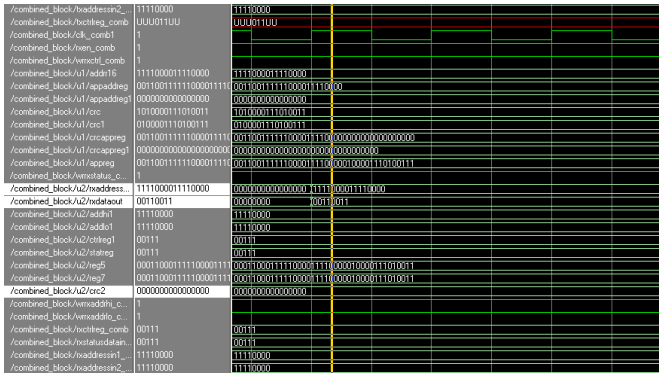

Fig. 5.4: Simulation result for O/P at the receiver for 16-bit address and 16-bit data

Simulation result for 8-bit data, 8-bit address and 32-bit CRC Check

Simulation result for crc-32

For the data<=11110000, 16bitaddress i.e. txaddressinlo<=11110000, we make clock=1, reset=0, wrtaddresshi=0 and wrtaddresslo=1. After the address and the data are attached together, we divide them with a constant polynomial of 32 bits and append the remainder of the division along the data and address. The simulation result for the generation of crc32 is highlighted below:

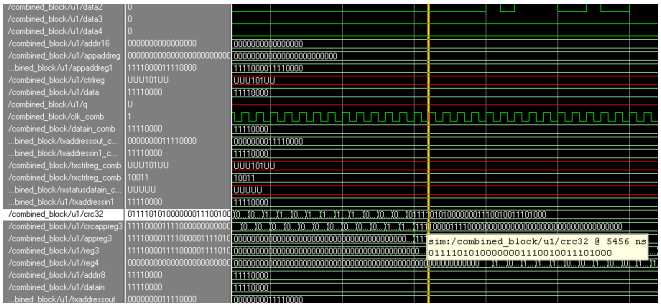

Fig.5.5-Simulation result for crc-32 of 8-bit address and 8-bit data

5.3.2 Simulation result of the final O/P at the receiver end for 8 bit data, 8-bit

Fig. 5.5: Simulation result for O/P at the receiver for 32-bit address and 32-bit data

Simulation result of the final O/P at the receiver end for 8-bit data, 8-bit address and 32-bit crc address and 32-bit crc

The highlighted simulation results shown below shows the result of the CRC check at the O/P which is crc32<=00000000000000000000000000000000 which indicates an error free transmission. The resultant address, data at the O/P which is same as that of transmitter is. rxdatout<=11110000 and rxaddressout<=0000000011110000.

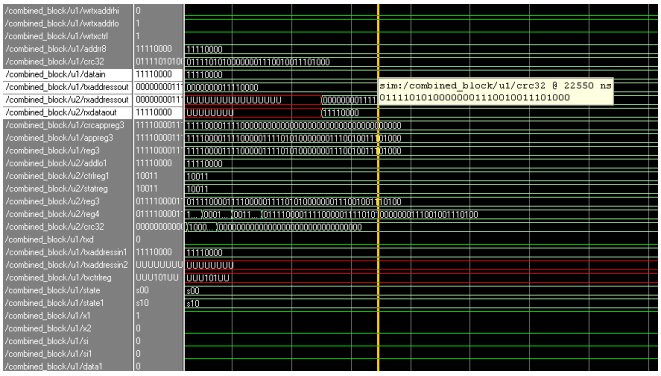

Fig. 6: Simulation result for O/P at the receiver for 8-bit address and 8-bit data and 32-bit crc

Simulation result for 8-bit data, 16-bit address and 32-bit CRC Check

Simulation result for 8-bit data, 16-bit address and crc-32

For the data<=11110000 and 16bitaddress i.e. txaddressinlo<=11110000, txaddressinhi<=11110000, txadrressout<=1111000011110000 we make clock=1, reset=0, wrtaddresshi=1 and wrtaddresslo=1. After the address and the data are attached together, we divide them with a constant polynomial of 32 bits and append the remainder of the division along the data and address. The simulation result for the generation of crc1 is highlighted below:

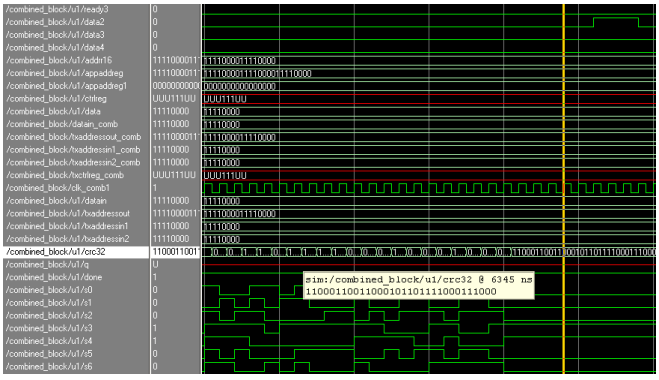

Fig. 7: Simulation result for crc-32 of 16-bit address and 8-bit data

Simulation result of the final O/P at the receiver end for 8-bit data, 16-bit address and 32-bit crc

The highlighted simulation results shown below shows the result of the CRC check at the O/P which is crc32<=00000000000000000000000000000000 which indicates an error free transmission. The resultant address, data at the O/P which is same as that of transmitter is. Rxdatout<=11110000 and rxaddressout<=1111000011110000.

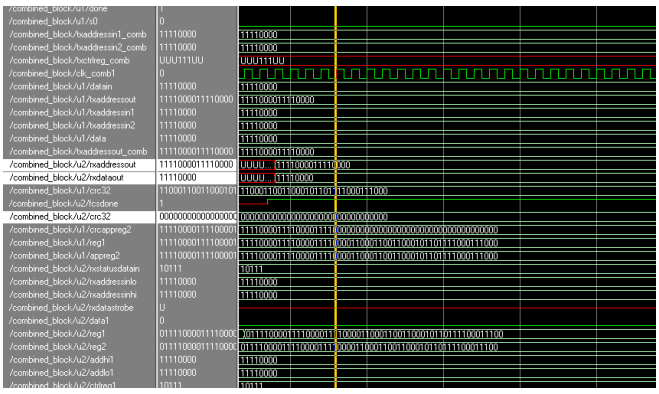

Fig. 5.8: Simulation result for O/P at the receiver for 16-bit address and 8-bit data and 32-bit crc

VI. CONCLUSIONS AND FUTURE SCOPE

Conclusions

HDLC Controller has been developed with the following features-

- The controller has the capability to operate in full duplex and half duplex mode.
- It can automatically check frame sequence generation using cyclic redundancy check i.e. CRC-16 and CRC-32 to ensure error free transmission.
- It is compatible with all the protocols present at the physical layer i.e.X.25 protocol and network layer i.e. Internet protocol (IP protocol).
- It has the capability to work in various modes Normal mode

Asynchronous response mode.

- 8-bit/16-bit address insertion and detection – selectable.
- To guarantee that a flag does not appear inadvertently anywhere else in the frame, HDLC uses a process called bit stuffing where it stuffs a zero whenever it finds 5 consecutive 1's.
- A VHDL model for the HDLC Controller has also been developed.

Future Scope

Although the HDLC Controller has been successfully developed in this thesis work, but some more features can be added to increase its utility. There are several modifications, which can further improve its performance. These are listed below-

- It can be further enhanced to256 Independent, Bi-directional HDLC Channels.
- Large 16kB FIFO in Both Receive and Transmit Directions.
- Transmit Packet Priority Setting.
- 16 physical ports (16 Tx and 16 Rx) that can be independently configured for channelized or unchannelized operation.
- Independent watchdog timer Facility to disable protocol functions

REFERENCES

[1] N. Mrudula1, K. Babulu2, "FPGA Implementation of HDLC Transmitter and Receiver," *IEEE International Conference on Communications, Circuits and Systems*, 2016.

[2] Ku. Rupal P. Bende, Prof. A. P. Bagade, Prof. S. R. Salwe, "Review on Design of HDLC Protocol using HDL", *International Journal of Innovative Research In Electrical, Electronics, Instrumentation And Control Engineering*, Vol. 4, Issue 2, February 2016.

[3] Hichem Semira, Mohamed Benouaret, Saliha Harize, "Implementation of a Single-Channel HDLC Controller on FPGA", *International Journal of Computer Applications* (0975-8887), Vol.-131 – No.3, December2015.

[4] Sarika G. Joshi, Vaishali S. Dhongde, Prof. Mrs.Mansi Wargantwar, "HDLC Protocol Implementation Using VHDL" *International Journal of Innovative Research in Science, Engineering and Technology*, Vol.-3, Special Issue 4, April 2014.

[5] Neeraj Kumar Mishra, "Xilinx HDLC Bit Stuffed Algorithm for Insertion and Deletion and Checking 32bit CRC for 16-bit Address", *International Journal of Research Review in Engineering Science and Technology* (ISSN 2278- 6643) | Vol.-2 Issue-1, March 2013.

[6] Gaurav Chandil1, Priyanka Mishra2, "Design and Implementation of HDLC Controller by Using Crc-16", *International Journal of Modern Engineering Research (IJMER)*, Vol.-3, Issue. 1, pp-12-18, Jan.-Feb. 2013.

[7] S. D. Samudra," FPGA Based HDLC Controllers: Comparative Review" IJCST Vol. 4, Iss ue 1, Jan - March 2013.

[8] Gaurav Chandil, Priyanka Mishra," Study and Performance Evaluation of Xilinx HDLC Controller and FCS Calculator" IOSR Journal of Engineering (IOSRJEN) Vol.- 2, Issue, PP 41-50, 10 October 2012.

[9] Ms. Kshitija S. Patil, Prof. G.D. Salunke, Mrs. Bhavana L. Mahajan, Dr. A.S. Hiwale, "Implementation of HDLC Protocol Using FPGA", *International Journal of Engineering Science & Advanced Technology [IJESAT]*, Vol.-2, Issue-4, pp. 1122-1131, Jul-Aug 2012.

[10] K. Sakthidasan, Mohammed Mahommed," Design of HDLC Controller Using VHDL" International Journal of Scientific & Engineering Research, Vol.-2, Issue 3, March-2011.

[11] Syed Manzoor Qasim and Shuja A. Abbasi, "FPGA Implementation of a Single-Channel HDLCLayer-2 Protocol Transmitter using VHDL", *International Conference on Electrical, Electronics and System Engineering*, 2003.

[12] S. Hamed Javadi and Ali Peiravi, "Design and Implementation of a High Bit Rate HDLCTransceiver Based on a Modified MT8952B Controller", Australian Journal of Basic and Applied Sciences, 2009.

[13] K. Sakthidasan, Mohammed Mahommed, "Design of HDLC Controller Using VHDL", International Journal of Scientific & Engineering Research, Vol.-2, Issue 3, March-2011.

[14] Harpreet Singh, Navneet Kaur, Vinay Chopra and Dr. Amardeep Singh, "Optimization of multi - channel HDLC protocol transceiver using Verilog", *International Journal of Computer Science Issues*, Vol.- 9, Issue 2, No 2, March 2012.

[15] Armaan Hasan Nagpurwala, Sundaresan C, Chaitanya CVS, "Implementation of HDLC Controller Design using Verilog HDL", *International Conference on Electrical, Electronics and System Engineering*, 2013.