ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

Data Communication Using HDLC Protocol

Shashank Rampelly[1], Santhosh Rao Seri[2], Gnaneshwara Chary[3], Krishanm Raju[4]

UG Student BVRIT, Narsapur, Medak[1,2]

Assistant Professor, BVRIT, Research Scholars, JNTUH, Andhra Pradesh, India[3]

Associate Professor, BVRIT, Research Scholars, JNTUH, Andhra Pradesh, India[4]

Abstract: The paper consists of two modes of communication i.e transmission and reception of a 8-bit data stream. Here the stream is generated has HDLC frames for the transmission by using 32:1 multiplexer, FIFO system, RLE Compressor technique and HDLC Framer. The RLE Compressor plays an important role in compressing the repetitive data. Similarly the data is retrieved from the HDLC Frames through a HDLC De-framer, RLE De-compressor and a De-multiplexer. Thus a data is moved from source to destination. This is a half duplex type of communication. This is initiated to make a quick and quality information flow.

Keywords: Channel Multiplexer, Channel Counter, FIFO System, RLE Compressor, HDLC framer, HDLC De-framer, RLE De-Compressor, De-Multiplexer

I. INTRODUCTION

The Data communications and networking are changing the way we do business and the way we live. Business decisions have to be made ever more quickly, and the decision makers require immediate access to accurate information. But before we ask how quickly we can get hooked up, we need to know how networks operate, what types of technologies are available, and which design best fills our mode of communication. This paper addresses four issues: data compression and de-compression, data transmission, data reception and HDLC Protocol framing and deframing. The 8-bit data stream is first introduced in the channels at 32:1 Multiplexer, the data is taken according to the channel selection. The channel selection is produced by the 5-bit channel counter. Then the data is shifted to the FIFO(First-On-First-Out) system. Here the data which enters first will exits first. The main purpose of using the FIFO system is to organize and temporary waiting for the Compressor call. Then the data is shifted from the FIFO system to a RLE(Run Length Encoding) compressor which is a compression technique where the repetitive data is compressed to a count and data i.e., the 8-bit data is added a 8-bit count before it so from here the data is modified to count added data[1]. The High-Level Data Link Control (HDLC) is a bit- oriented code-transparent synchronous data link layer protocol developed by the International Organization for Standardization (ISO). This is situated between the data link layer and physical layer in OSI Layers Here the data is framed at transmission and de-framed at reception parts. The un-disturbed, long distance transmission possibility makes it unique and advantageous compared to other protocols. A compression technique where the runs of repeated data is compressed into count and data to save the lots of memory is adapted to this HDLC Protocol to make a best way of wire-less data flow. At the receiver block the De-framing, De-compressing and data retrieving is done.

II. DATA COMPRESSION SYSTEM

The data compression is followed to remove irrelevant data in a bulk original data. The compression is effective when the measured data rate are expected to be slow and repeatable. The task of this unit is crucial to the system power performance[2]. RLE is a conceptually simple form of compression. RLE consists of the process of searching for repeated runs of a single symbol in an input stream, and replacing them by a single instance of the symbol and a run count. RLE compression is only efficient with files that contain lots of repetitive data. These can be text files if they contain lots of spaces for indenting but line-art images that contain large white or black areas are far more suitable. Computer generated color images (e.g. architectural drawings) can also give fair compression ratios[12].

III. DATA COMMUNICATION

It has 32 bit multiplexer, channel counter, FIFO (first in first out), RLE Compressor and HDLC Framing at the transmission end. A 32 bit de-multiplexer, FIFO(first in first out),RLE De-compressor and a HDLC De-framing at the receiving end.
The architecture is shown below.

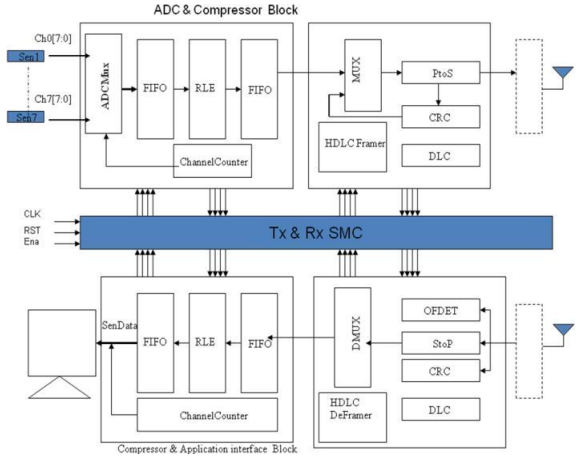

Figure 3.1. Architecture of Data communication.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

IV. DATA TRANSMITION

CHANNEL MULTIPLEXER

To select the 32 Channels information we are using the multiplexer, it consist of 8-bit data with each channel. To select each channel we are using Channel counter as a Selection line. This Multiplexer consist of Enable signal and Data (Input) signal, where the Data signal shows the Output Information of the Multiplexer. When En=1, depends upon the channel selection input channel information transmits to Output.

CHANNEL COUNTER

A channel counter is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal.

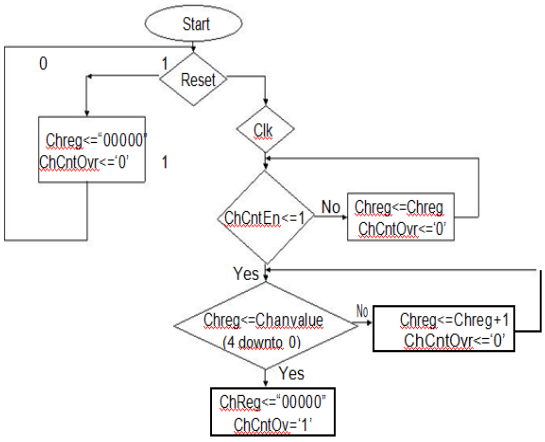

Figure 4.1. Channel Counter

- Channel counter consist of Clk signal, Rst signal, Enable signal, Channel value, Channel selection and channel CntOvr.
- This Channel counter is used to select the channel value for the Data transmission from the Multiplexer.
- When Rst='0', ChReg is empty (00000) and the
- CntOvr shows that no count is there nothing but it shows zero (0). After completion of the counting the CntOvr shows one (1).
- When Channel counter counting the value (Clk='1',
- En='1') ChReg will be increments and at that time the CntOvr shows zero (1)[3].

FIRST IN FIRST OUT (FIFO)

- FIFOs provide solutions for a wide variety of data buffering needs, including high-speed data acquisition, multiprocessor interfaces, and communications buffering. FIFO Logic diagram is shown below. Input ports are controlled by a free running clock write(CKW)
- (CKW) and a write-enable pin ENW. When ENW is asserted, data is written into the FIFO on the rising edge of the CKW signal.

While ENW is held active, data is continually written into the FIFO on each CKW cycle.

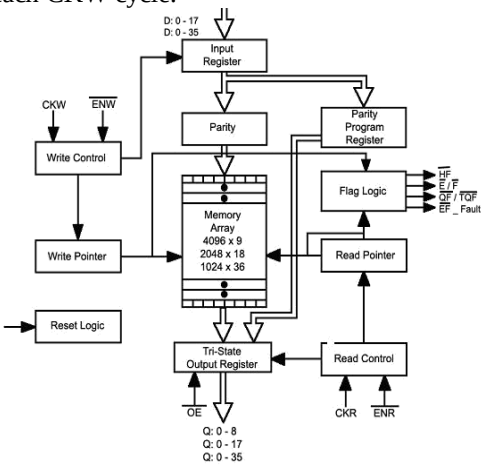

Figure 4.2. Logic diagram of FIFO

The output port is controlled in a similar manner by a free running read clock (CKR) and a read enable pin (ENR). In addition, the three FIFOs have an output enable pin (OE) and a master reset pin (MR). The read (CKR) and write (CKW) clocks may be tied together for single-clock operation or the two clocks may be run independently for asynchronous read/write applications. Clock frequencies up to 28 MHz are achievable in the three configurations. The FIFO designed for use in systems operating in radiation environments[5].

- FIFO is used to store the data which is transmitted from the Multiplexer.
- FIFO consists of WrClk, RdClk, Rst, DataI, FIFOEna, FIFOWr, FIFORd, Data, Full, Empty.
- First inserted data is Writable into the FIFO and it first Readable from the FIFO.
- FIFO depends upon the Write signal and Read signal. The inserted data is taken by Data In signal and Output information is taken by the Data out signal.
- When FIFO Full means the length of the FIFO is inserted by total channel information and the "Full" signal shows the '1'.
- If the information is already readed then it shows the Empty of the FIFO and "Empty" signal shows '1'.
- When WPtr-RPtr=31 then Full=1. It shows that the RPtr at First data so it shows the total information is filled in the FIFO that's why the signal Full shows '1'.
- When WPtr-RPtr=0 then Empty=1. It shows that RPtr at 31st channel means the reading of the information is completed .So the FIFO is now Empty and the signal Empty shows '1'.

RLE COMPRESSOR:

In this compression block the input data is to give to the

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

"INPUT FIFO BLOCK". This block takes the input data in first in first out sequence and then it is applied to the "COMPRESSION BLOCK". This compression block compresses the input digital data which can decrease the total size of the input. Decreasing of the input data is done by compressing the given data in which the bits that has a difference one to each and to eliminate the one bit. After compression the compressed data will go to "OUTPUT FIFO BLOCK". This block will give the reduced compressed output data. This compressed data is then applied to the "DECOMPRESSOR BLOCK". In "RLE Compressor block" can compress the input data. In initializing the flow chart to start and consider the register 1 and register 2.The input byte is to be read when it the reset signal is enable it starts to read the data and then write it. If next input will come it will store the next FIFO address and then to decrease the count and the final output is decompressed like reverse order of compression method. When the total address is filling it is not possible to store the next input so that the out is "Data full". When the data full that input data is to be carry forward then the FIFO is in "Idle" state then the next input will store. When the second bit enters into FIFO block it compare it"s pervious bit value, if both bits are having same value then t then he count increase by one .Example: If 6 bytes are came continuously by 8 times then it will represent as like a function (6, 8),if 4 bytes are repeated for 5 times then it will represent as (4,5).by using this functional logic the continues data is to be count by its number of times then the compressed data stores in output FIFO. In this block is modified output data which gets compressed file which reduces the file size.

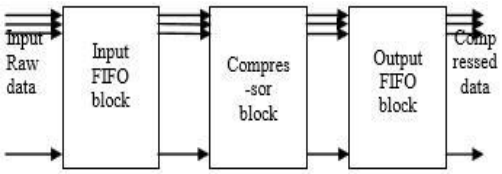

Figure 4.3. RLE Compressor Block diagram

HIGH-LEVEL DATA LINK CONTROL (HDLC)

It is a bit-oriented code-transparent synchronous data link layer protocol developed by the international organization for standardization (ISO)[4]. The current standard for HDLC is ISO 13239. Provides connect-oriented and connection less service[13].

HDLC FRAME

The basic structure of the HDLC frame is shown in Figure.

Figure 4.4. Main fields of the HDLC frame structure

The HDLC frame fields are working as follows

a. Start and end flags:

Start and end flags, represented by the sequence (01111110), are required for synchronous transmission.

b. Address field:

Address field is used to identify the destination address at the receiver side.

c. Control field:

Control field is used to classify the HDLC frames according to the link configuration type.

d. Information field:

Information field contains the transported data. e. Frame check sum (FCS): Frame check sum (FCS) is used to detect errors by adopting CRC generation[6].

HDLC TRANSMITTER

The main blocks of HDLC transmitter contains a. State machine: An FSM is responsible for generating all the necessary internal control signals required by the different modules.

a. Transmitter controller:

The controller checks if there is a valid data output from the compressor and then starts loading the bytes into FIFO memory storage. Then the data are read serially from the memory storage and sent to the CRC module.

b. FCS Generator:

The FCS Generator is used to generate the frame check sequence (FCS).

c. Bit stuffer:

The bit stuffer is responsible for examining the frame content and checking every five consecutive 1s bits including FCS bits. If five consecutive 1s are detected, a 0-bit is inserted into the serial bit stream. This helps the receiver to distinguish the actual data transmitted[10]

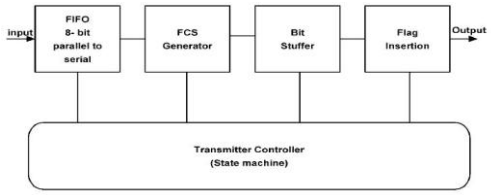

Figure 4.5. HDLC transmitter control block diagram

d. Flag insertion

The start and end flags are generated at the final stage and attached to the frame. Also, the transmitter fills the gaps between the frames when the transmission is idle by sending a sequence of eight consecutive 1s. The received frames are processed inversely by similar structure at the receiver control to recover the transmitted bytes. All the modules of both HDLC transmitter and receiver have been modeled, simulated and synthesized successfully.

TRANSMITTER SHIFT REGISTER:

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

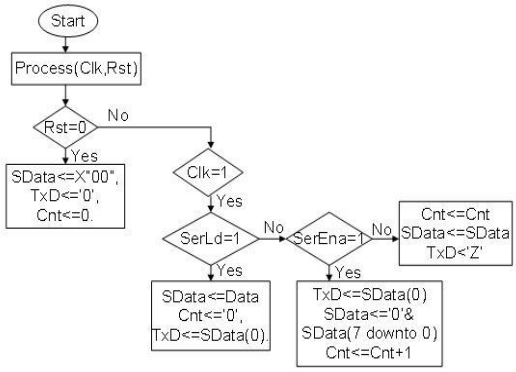

Figure 4.6. Flow chart of Transmitter shift register

In this Parallel To Serial shift Register takes the information from the FIFO and it converts it into serial data. It consists of Data as input and transmitted Data as Output. When the Shift is completed SerOvr shows zero (0). When Clk='1' and SeriD is Enable the output takes the Data from input in the manner of shifting format. At this time SerlOvr=1.

CYCLIC REDUNDANCY CHECK (CRC):

Error detection is important whenever there is a non-zero chance of your data getting corrupted. Whether it's an Ethernet packet or a file under the control of your application, you can add a piece of redundant information to validate it. The simplest example is a parity bit. Many computers use one parity bit per byte of memory. Every time the byte gets written, the computer counts the number of non-zero bits in it. If the number is even, it sets the ninth parity bit, otherwise it clears it. When reading the byte, the computer counts the number of non-zero bits in the byte, plus the parity bit. If any of the nine bits is flipped, the sum will be odd and the computer will halt with a memory error. (Of course, if two bits are flipped--a much rarer occurrence--this system will not detect it.)[11]

For messages longer than one byte, you'd like to store more than one bit of redundant information. You might, for instance, calculate a checksum. Just add together all the bytes in the message and append (or store somewhere else) the sum. Usually the sum is truncated to, say, 32 bits. This system will detect many types of corruption with a reasonable probability. It will, however, fail badly when the message is modified by inverting or swapping groups of bytes. Also, it will fail when you add or remove null bytes. Calculating a *Cyclic Redundancy Check* is a much more robust error checking algorithm. In this article I will sketch the mathematical foundations of the CRC calculation and describe two C++ implementations--first the slow but simple one, then the more optimized one. The CRC is a very powerful but easily implemented technique to obtain data reliability. The CRC technique is used to protect blocks of data called Frames. Using this technique, the transmitter appends an extra n- bit sequence to every frame called Frame Check Sequence (FCS). The FCS holds redundant information about the frame that helps the transmitter detect errors in the frame. The CRC is one of the most used techniques for error detection in data communications. The technique gained its popularity because it combines three advantages:

- Extreme error detection capabilities.
- Little overhead.
- Ease of implementation

The CRC algorithm works above the binary field. The algorithm treats all bit streams as binary polynomials. Given the original frame, the transmitter generates the FCS for that frame. The FCS is generated so that the resulting frame (the cascade of the original frame and the FCS), is exactly devisable by some pre-defined polynomial. This pre-defined polynomial is called the devisor or CRC Polynomial.

V. DATA RECEPTION

HDLC DE-FRAMER:

A high speed multi channel HDLC de-framing machine receives a series of code words from a FIFO queue. For each code word received, a corresponding channel is identified. HDLC state, the code word to be de-framed and, a mask are loaded into registers, and the de-framer is activated.

The de-framer returns an updated HDLC state, status flags, and potentially a de-framed data word[14].

- Data received 01111110000111011111011111011001111110
- After de-stuffing and de-framing 000111011111-11111-110
- A high speed multi channel HDLC framing machine cycles through Time Division Multiplexed (TDM) channels identifying a channel table for each channel.
- HDLC state, data to be framed, and a mask are loaded into registers and the de-framer is activated. The de-framer returns an updated HDLC state, status flags, and a framed code word.
- The framed code word is multiplexed in a FIFO queue for output on the high speed TDM line. A high speed multi channel HDLC de-framing machine receives a series of code words from a FIFO queue.
- For each code word received, a corresponding channel identified. HDLC state, the code word to be de-framed and, a mask re loaded into registers and the de-framer is activated.

The de-framer returns an updated HDLC state, status flags, and potentially a de-framed data word. Commercially available products, such as the product identified by the trademark "DC2K-FR" and manufactured by Racal-Air tech Limited, a U.K. company, "de-stuff" each frame (i.e., removes the zero bits added by the stuffing process), transform it (e.g., by encryption--when required) and re-stuff it before transmitting. Since the data being re-stuffed has been transformed, it may have very different statistical characteristics to the original

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

frame, and a different number of bits may be added by the re stuffing process. In these commercially available products, an "equalizing" First-in First-out device (FIFO) is used to minimize the loss and corruption of frames due to this effect. In the equalizing FIFO, the next byte to be retrieved is the byte that has been in the queue for the longest time. The equalizing FIFO used is fairly large, having to cope with potentially very different input and output data rates. Encrypted data is essentially random, whereas the input data that is dyestuff can be highly formatted[7].

RLE DE-COMPRESSOR:

In the "De-Compressor block" the input which takes from the "Compressor block"is applied to the "Input FIFO". In "RLE decompression block" can extract the compressed output.. In decompression of the compressed data flows to start and consider the register 1 and register 2.The input byte is to be read when it the reset signal is enable it starts to read the data and then write it. If next input will come it will store the next FIFO address and then to decrease the count and the final output is decompressed. When the total address is filling then it is in ideal state, then at the out of the output FIFO to get the original input.

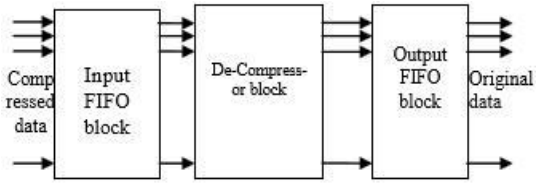

Figure 5.1. RLE De-Compressor Block diagram

In this block, the sequential input decompresses the input in "De-Compressor Block" and that data which decompressed is to given the "Output FIFO". When the first input is entering into the decompression block then input values are decompress by basing of its functional value. if the compressed input is (6, 8),that means 6 is repeated by 8 times, so that the out of decompressed block will is to print 6 for 8 times by comparing its functional value and to decrease the count by using counter signal and the remaining functional values are also be decompress as same passion[8].

DE-MUTIPLEXER:

As the multiplexer selects the data from 32 channels to get one out of them, de-multiplexer will do a just opposite operation. It arranges the data in 32 channels according to the selection lines. When En=1, depends upon the channel selection input, data is retrieved in 8 bits to the output.

VI. RESULTS AND ANALYSYS

1. Transmitter simulation results

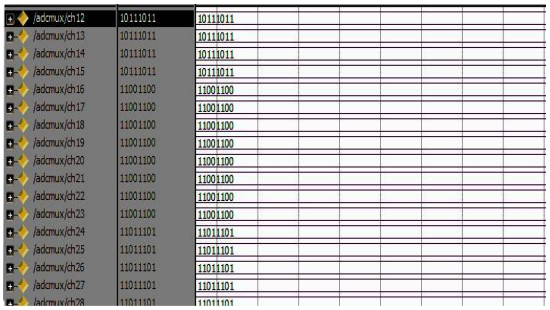

Figure 6.1. MUX Result

The 32 channels information is taken as input and the selection of the information is depends up on the channel selection line. If channel selection is 00001 then it reads the data of the second channel i.e 10101010.this is similar to all 32 channels[15].

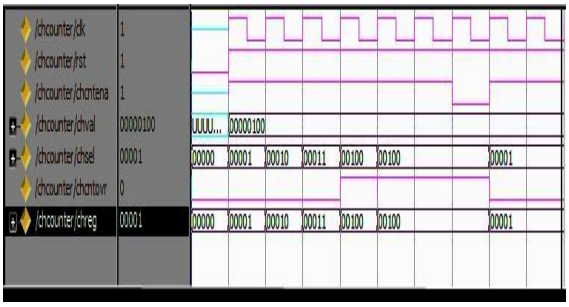

Figure 6.2. Channel counter result

Rst and clk are at enable then the input of the channel counter is any value from 32 channels (ex: 01000) it shows that after 8 clock pulses the cntovr is '1'.here the channel value is equal to register value.

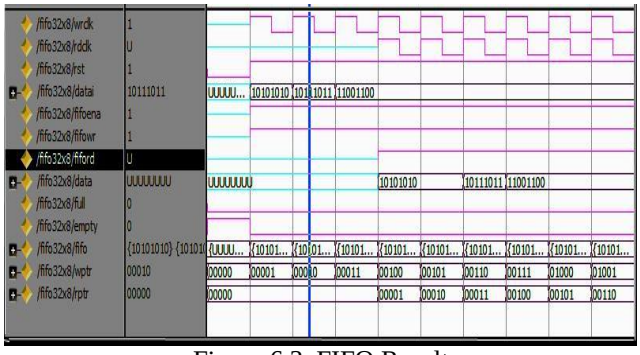

Figure 6.3. FIFO Result

To write the data in the fifo we should enable the write clk and at same time fifoena signal and fifo write signal both will be active high then the input of the data is written in to the fifo. To read the data from the fifo enable the fiford signal then the already written data will be readable through the data out signal. After 32 channels information written then the signal full shows '1'. After reading the total information the empty signal shows '1'.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

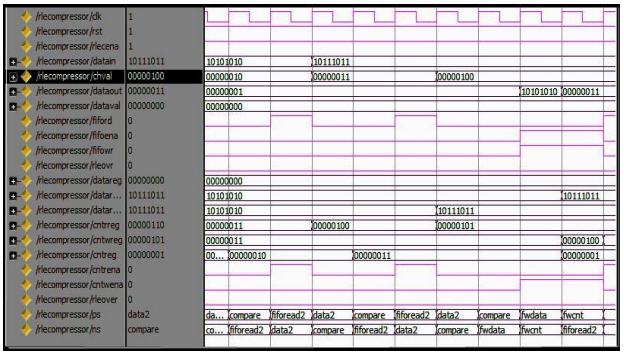

Figure 6.4. RLE compressor result

compare it if not then it gives the first data information and how many times it repeated.

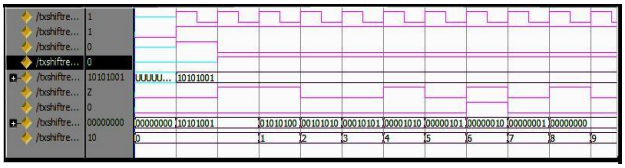

Figure 6.5. Transmitter Shift register result

To convert parallel data to serial data this shift register is useful. The shifting of data depends up on the serld signal .When it is zero it adds '0' to the given data from LSB. It will work up to final data will be "00000000".

2. Receiver simulation results:

Figure 6.6. De multiplexer wave form

The output is depends up on the selection line. In above results 01011 channel consist of data AA.Nothing but 10101010.

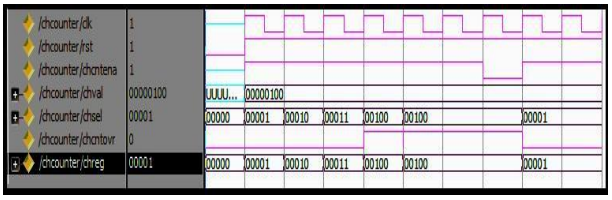

Figure 6.7. Channel Counter wave form

Rst and clk are at enable then the input of the channel counter is any value from 32 channels (ex: 01000) it shows that after 8 clock pulses the cntovr is '1'.here the channel value is equal to register value.

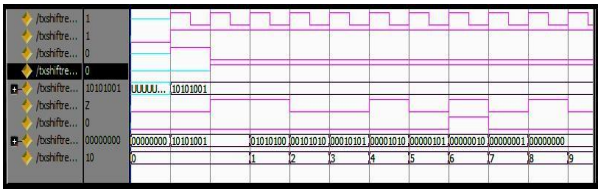

Figure 6.8. Shift registers wave form

In shift register the 0's will be shifted then the data 00001111 will be modified as 00000000.Here 8 times 0 will be shifted.

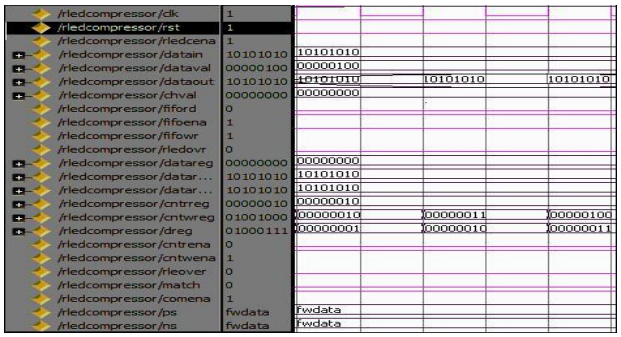

Figure 6.9.RLE decompressed wave form

In this data is comes depends up on the count value. Here count value is 4.So the data value 10101010 Repeated 4 times.
Thus the original data is received and used.

VII. CONCLUSION AND FUTURE SCOPE

CONCLUSION:

In this work a software simulation environments have been used to model a data compression system. Although HDL provides many high-level abstractions and language constructions for behavioral modeling, its synthesizable subset is far too restrictive for system-level design. The two main operations implemented by the transmitter FPGA are compression and framing. RLE has been used to compress the stored data bytes efficiently with an optimal number of states. The HDLC protocol has been used successfully for framing the data and providing error-handling mechanism to the receiver. All the FPGA modules have been verified and implemented using Xilinx Spartan-3 device. During the development of the VHDL codes, our main goal was to make all the units synthesizable[9].

FUTURE SCOPE:

In this work up to N number of channels can be extend to transmit the more data. If this data is compressed then more power is saved and as per the applications we can replace the data compression system.

ACKNOWLEDGEMENT

The authors would like to thank Mr. I.A.Pasha, HOD, ECE Department, BVRIT and Sanjey Dubey, Professor BVRIT, Narsapur, Medak for his valuable suggestions And they also express thanks to parents, friends and colleagues.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue 5, May 2014

REFERENCES

[1] Anderson.J., Rappaport.T.S., and Yoshida.S., "Propagation measurements and models for wireless communications channels"., Proceeding IEEE communication Management, Volume I., paper(42-49), 1995.

[2] Andreasson,J, Ekstrom,M., Fard.A., Castano.J.G., and Johnson.T., "Remote system for Patient Monitoring using Bluetooth/SPI Trade"., Proceeding of IEEE Sensors, volume noI paper (304-307),June 2002.

[3] Banitasa,K.A., Istepanian, R.S.H., Tachakra, S., and Owens, T.J, "Modeling issues of wireless LANs for accident and emergency departments"., Proceeding 0f IEEE/EMBS 23rd Annual International Conference, Turky, volume IV, page(3540-3543), October 2001.

[4] Bin.H., Zhigong.W., LuFeng.Q., and Yuanline.L., "Design of a multi-channel high sped FIFO applied to HDLC processor based on PCI bus"., Proceeding of IEEE Circuits and Systems and West Sino Expositions, China, Volume II. paper(1476-1480), June-July 2002.

[5] Blackner.A., Albl.S., and Acherr.W., "Configurable computing architectures for wireless and software defined radio-a FPGA prototyping experience using high level design-too-chains"., Proceeding of IEEE System- on-Chip Conference, USA, paper(111-116), September 2004.

[6] Bateman.A., and Haines.D.M., "Direct conversion transceiver design for compact low-cost portable mobile radio terminals"., Proceeding of conference on IEEE Vehicular Technology VTC, paper(57-62), May 1989.

[7] Darrel,R.J, 'Data synchronization simulation using the math works'. Proceeding IEEE conference on Communications, USA, paper(706-710), June 1996.

[8] Folke.M., Andreasson,J., Hok,B., Linden,M., Nilsson,G.,Ekstrom,M., and Backlund. Y, "Artefact reduction in telemetric ECG monitoring",Proceeding of 12th Nordic Baltic Conference., on Biomedical Engineering and Medical Physics,Island,volume II, paper(18-22) June 2002.

[9] Kim,Y., kim,K.,Shim,Y.,Ahn,T.,Sung,W.,Choy,K.and Has., "An integrated Hardware-software co- simulation environment for Heterogeneous system prototyping"., Proceeding of 7th IEEE international workshop Rapid system prototyping(RSP'96),Greece, paper(66-77),June 1996.

[10] Michel.R., and Patel.A., "Performance analysis of an HDLC based protocol with data compression"., Proceeding computer communication, Volume VII., paper(567-575),1997

[11] Qasim, S.M., and Abbasi.S., "FPGA Implementation of a single-channel HDLC layer-2 protocol transmitter-using VHDL"., proceeding of 15th International conference on Microelectonics ICM, Cario,Egypt, paper(265-268), December 2003.

[12] Shoufeng.M., and Michael,L., "Mixed-signal modeling using Simulink based C"., Proceeding of conference on IEEE Behavioural and Modeling and Simulation (BMAS),San Jose, USA, paper(128-133), September 2005.

[13] Shyamprakash.A., and Ravikumar.C.P., "Verilog modeling and simulation of a communication coprocessor for multicomputers"., Proceeding Conference on IEEE Verilog HDL,paper(58-66), March 1995.

[14] Steycrt.M.,Borremans.M.,Janssens.J.,DeMuter.B.,Cra inckx.J.,Crols.J.,Morifuji.E. Momo se.S., and Sansen.W., "A single-chip CMOS transceiver for DCS-1800 wireless communications"., Digest of Technical Papers IEEE ISSCC, Sanfrancisco, USA, paper(48-49), Februry 1998.

[15] Yuanlin,L., Zhigong.W., Lufeng.Q., and Bin.H., " Communications Design and implementation of multi-channel high speed data processor"., Proceeding IEEE Circuits and Systems and West Sino Expositions, China, Volume II., paper(1471-1475), June-July 2002.