Implementation of Run Length Encoding on FPGA Spartan 3E

[1]Rakesh Karmakar, [2]Rajib Sarkar
[1]Dept. of CSE, Tripura University (Central University), Tripura, India
[2]Dept. of ETE, MBES COE, Ambajogai, Maharashtra, India

Abstract
In computer science, compression algorithms are an important technique to reduce the original data bits into lesser number of bits. Lossless compression is a special class of data compression algorithm involves in reducing bits by identifying and eliminating statistical redundancy. A simple scheme that provides good lossless compression of data containing lots of runs of the same value is Run Length Encoding. In this paper a technique to model the run length encoding algorithm on reconfigurable platform has been described. FPGA implementation result shows that the circuit requires very small amount of digital hardware. Estimated power consumption and maximum operating frequency of the circuitry is also provided.

Keywords
Run Length Encoding, Image Compression, FPGA, Lossless Compression, Decompression, Entropy Coding

I. Introduction

In computer system, compression technique is one of the most important aspects to increase system performance. To perform lossless data compression run length encoding RLE [1] is well suited.

There are some related works to Run Length Encoding available in literature. Scott Hauck et. al. [2] proposed a paper on Run length compression techniques for FPGA configurations. The paper covers a technique by which files can be compressed by a factor of 3.6 times. Amritpal Singh et. al [3] proposed an enhanced run length coding for JPEG image compression. In that paper a technique to remove the unintended RUN, LEVEL (1, 0) pair used for a single zero present between the two non-zero characters was presented. So instead of using (1, 0) pair for the zero between non-zero characters, a single '0' will be encoded. Samir Kumar Bandhyopadhay et. al.[4] proposed a paper on image compression using approximate matching and Run Length. The work concentrates on the lossless compression of image using approximate matching technique and run length encoding.

In this paper, a technique for the run length encoding using system c programming is modeled. The electronics system level ESL model is implemented on FPGA platform in the laboratory environment. Logic circuit requirement and FPGA resource utilization is presented. The estimated power consumption of the implementation is found to be 56 mW which is very suitable for battery power application. The resource utilization report shows that very small amount of FPGA resources are used keeping ample scope to implement the rest of the circuitry in the same FPGA.

This paper is organized as follows. Section II briefly describes the data compression model. Section III describes the information content and entropy which mainly demonstrates how the information is measured. Section IV describes run length encoding. The sub-section A features the compression methodology while the sub-section B describes the algorithm. The design and implementation modeling has been described in section VI. Section VII describes the FPGA implementation and results. The paper is concluded in Section VIII.

II. Data Compression Model

Essentially, data compression deals with the means of reducing the number of bits required for storing and transferring information. Compressing data is necessary for a number of reasons. To start with, compressing data allows a user to keep more information in the system memory than otherwise possible. Every data compression technique uses a model to function in which the input stream generated from a data source is fed into an encoder. The encoder then encodes and compresses data. A notion of model defines the parameters that need to be used by the compression algorithm.

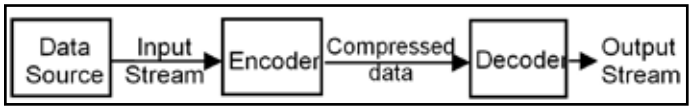

Fig. 1: Data Compression Model

To regenerate original data from the compressed data, decoder is used. The decoder applies the reverse algorithm of that used by the encoder. Moreover, the decoder has some prior knowledge as to how the data is being encoded. This is where standardized compression algorithms come into play.

III. Information Content and Entropy

Information is a measure of the probability of a message being selected from the set of all possible messages. Numerically, information is measured in bits. One bit is equivalent to the choice between two equally likely choices. When several choices are equally likely, the number of bits is equal to the logarithm of the number of choices taken to the base two. When the various choices are not equally probable, the situation is more complex. Information content of a message is measured in terms of its entropy. Entropy of a character is defined as the negative logarithm of the probability with which the character occurs in a message. Number of bits required to code a character = - log (probability).

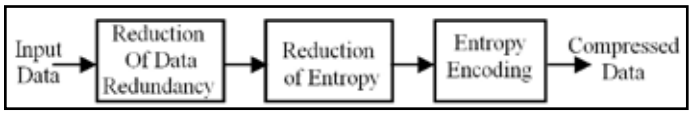

Fig. 2: Formation of Compressed Data

IV. Run Length Encoding

Run-length Encoding (RLE) is a very simple form of data compression in which runs of data that is, sequences in which the same data value occurs in many consecutive data elements are stored as a single data value and count, rather than as the original run. This is most useful on data that contains many such runs: for example, simple graphic images such as icons, line drawings, and animations. It is not useful with files that don't have many runs as it could greatly increase the file size.

A. Compression

RLE compresses sequences containing subsequent repetitions of the same character. By compressing a particular sequence, its code

is obtained. The idea is to replace repetitions of a given character (like aaaaa) with a counter saying how many repetitions there are. Namely, it is represented by a triple containing a repetition mark, the repeating character and an integer representing the number of repetitions. For example, aaaaa can be encoded as #a5 (where # represents the repetition mark).

A technique has been adopted to represent the alphabet, the repetition mark, and the counter. If the word consist of n characters represented by integers from the set $\Sigma = \{0, 1, \ldots, n-1\}$. The code of a sequence of characters from Σ is also a sequence of characters from Σ. At any moment, the repetition mark is represented by a character from Σ, denoted by e. Initially e is 0, but it may change during the coding.

The code is interpreted as follows:

1. Any character a in the code, except the repetition mark, represents itself,
2. If the repetition mark e occurs in the code, then the two following characters have special meaning:

- If e is followed by ek, then it represents k + 1 repetitions of e,
- Otherwise, if e is followed by b0 (where $b \neq e$), then b will be the repetition mark from that point on,
- Otherwise, if e is followed by bk (where $b \neq e$ and $k > 0$), then it represents k + 3 repetitions of b.

This scheme can be used to encode any sequence of characters. For instance, for n = 4, the sequence 100222222333330302020000 can be encoded as 10010230320100302101. First character of the code 1 means simply 1. Next 001 encodes 00. Then, 023 represents 222222, 032 represents 33333, and 010 switches the repetition mark to 1. Then 0302 represents itself and finally 101 encodes 0000. A sequence may be encoded in many ways and code length may vary.

B. Algorithm

Step 1. Set the previous symbol equal to an unmatchable value.
Step 2. Read the next symbol from the input stream.
Step 3. If the symbol is an EOF exit.
Step 4. Write out the current symbol.
Step 5. If the symbol is a does not match the previous symbol, set the previous symbol to the current symbol, and go to step 2.
Step 6. Read and count additional symbols until a non-matching symbol is found. This is the run length.
Step 7. Write out the run length.
Step 8. Write out the non-matching symbol.
Step 9. Set the previous symbol to the non-matching symbol, and go to step 2.

While implementing RLE, a little attention to detail is required in Step 6. The run length is stored in a finite number of bits. In this work an unsigned character is used. Runs longer than the amount that can be counted need to be broken up into to smaller runs. When the maximum count is reached, the count value is written and process of looking for a run starts all over again. It also needs to handle the case where a run is ended by an EOF. When a run is ended by an EOF, write out the run length and exit.

V. Implementation Design

The implementation of data compression algorithm-RLE is done on Spartan 3E development board using Xilinx Platform Studio 11.1. The work is divided in both hardware and software, while the hardware part was done using embedded development kit EDK [7] and the software part was done using system C programming language. The complete design of the RLE encoder on FPGA is presented.

Hardware design carried out using Base System Builder (BSB). The block diagram and system assembly view of the design is shown in the fig. 3 and 4 respectively. Xilinx ISE provides the unique functionality of each and every peripheral which we need during development. The system assembly view is shown in fig. 5.

In order to read and write data SDRAM IP (Intellectual Property) core to the design has been added. Data are given as input from the keyboard from the RS232 port through Hyper Terminal. RS232 peripheral has been introduced during the BSB design. The keyboard interfacing has been established with the Spartan 3E board using UARTlite [8] component of Intellectual Property IP core [9]. This UART is a minimal hardware implementation with minimal features. Most of the features, including baud rate, parity, and number of data bits are only configurable when the hardware device is built, rather than at run time by software.

VI. FPGA Implementation

The design is implemented in XA Spartan®-3E FPGA [10] platform. XPS synthesis report of the implementation is presented in Table 1. The devices used are adder, multiplexers, registers and counters. The mux chooses the output of the base register when the down-counter equals zero. It consists of a register to hold the current address to output; a down counter to count the length; an adder, to add the offsets to the previous value; and a mux to choose between a previous valued added to the offset and the new base value. When a new code word arrives, the base value is written into the address register at the same time that the length is written into the down counter. The down-counter then counts down until zero, while the address register captures its previous value plus the offset. This continues until the down-counter reaches zero.

Fig. 6 shows the encoded output for a string of input in the HyperTerminal. Table 2 shows the reference system address map. It has been found that the estimated power consumption using Xilinx power is very low and it is 56mW. The maximum operating frequency found is 117.08 MHz.

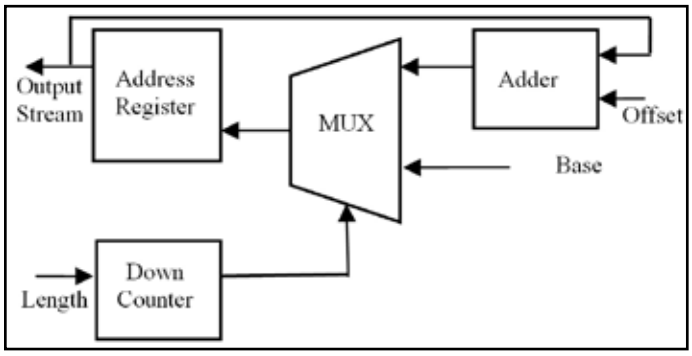

Fig. 3: Block Diagram for RLE Encoding

Fig. 4: Data Compression Model

ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

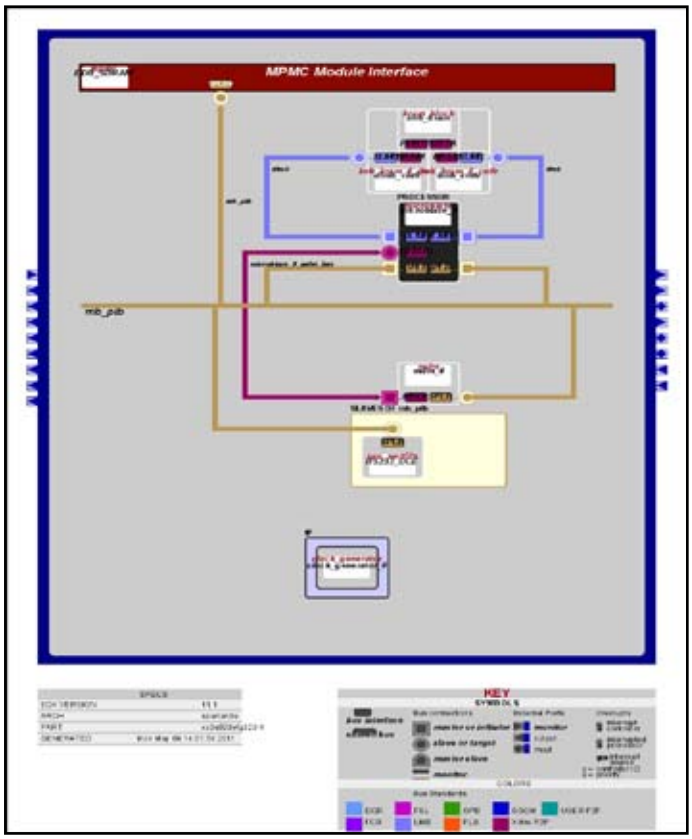

Fig. 5: System Assembly View

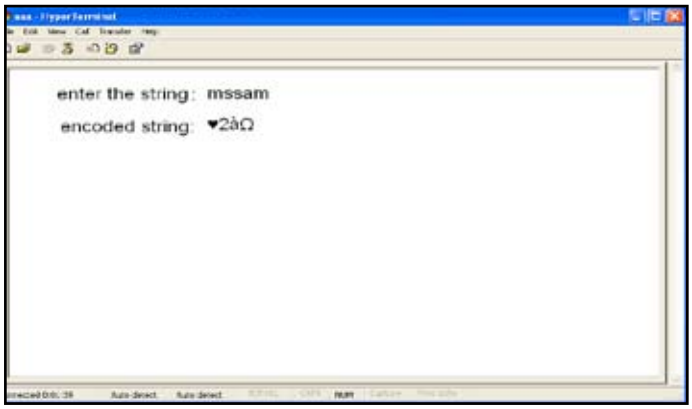

Fig. 6: Encoded String Output in Hyper Terminal

Table 1: XPS Synthesis Report

Report	Generated	Flip Flops Used	LUTs Used	BRAMS Used	Errors
System	Fri Mar 13 14:42:52 2011	1538	2216	16	0

Table 2: Reference System Address Map

Instance Peripheral	Base Address	High Address
debug_moduleopb_ mdm	0x41400000	0x4140FFFF
dlmb_cntlrlmb_bram_ if_cntlr	0x00000000	0x00001FFF
ilmb_cntlrlmb_bram_ if_cntlr	0x00000000	0x00001FFF
RS232_DTE opb_ uartlite	0x40600000	0x4060FFFF

VIII. Conclusion

A technique to model the data compression algorithm run length encoding using system c has been described in this paper. The electronics system level ESL model is implemented on FPGA using Xilinx Platform Studio 11.1. The FPGA implementation result shows that the design is attractive from resource utilization, power consumption and operating frequency point of view.

References

[1] Held, Gilbert,"Data Compression: Techniques and Applications, Hardware and Software Considerations", second edition, John Wiley & Sons, New York, NY, 1987.

[2] S. Hauck, W. D. Wilson,"Runlength compression techniques for FPGA configurations", In Proceedings IEEE Symp. Field-Program. Custom Comput. Mach., 1999, pp. 286–287.

[3] Amritpal S., V.P. Singh,"An Enhanced Run Length Coding for JPEG Image Compression", International Journal of Computer Applications, Vol. 72, No. 20, June 2013, pp. 21-26.

[4] Samir K. B., Tuhin U. P., Avishek R.,"Image Compression using Approximate Matching and Run Length", International Journal of Advanced Computer Science and Applications, Vol. 2, No. 6, 2011, pp. 117-121.

[5] Douglas L Perry,"VHDL Programming by Example", McGraw-Hill 4th Edition 2002.

[6] Pong P.,"RTL Hardware Design using VHDL", John Wiley & Sons publication 2006.

[7] Xilinx "EDK Concepts, Tools & Techniques", [Online] Available: http://www.xilinx.com//support/documentation/application_notes/

[8] Xilinx,"AXI UART Lite", [Online] Available:http://www.xilinx.com/products/intellectual-property/axi_uartlite.htm

[9] Xilinx "Intellectual Property", [Online] Available:http://www.xilinx.com/products/intellectual-property/

[10] Xilinx "XA Spartan-3E FPGA Family", [Online] Available: http://www.xilinx.com/products/silicon-devices/fpga/xa-spartan-3e/

Rakesh Karmakar received his Bachelor of Engineering degree in Computer Science and Engineering from SAIT under Visvesvaraya Technological University, Belguam in 2008 and M.Tech. in Computer Science Engineering in 2011, from Tripura University (A Central University). He has many years of industrial experience. His research interest includes VLSI, Embedded System and Database Management System.

Rajib Sarkar received his Bachelor of Engineering degree in Electronics & Telecommunication Engineering from Dr. BAM University, Aurangabad in the year 2005 and M.Tech. in Computer Science Engineering in 2010, from Tripura University (A Central University). His research interest includes VLSI, Embedded System and Computer Networking. He has published quite a few research papers in International & Indian Journals and conferences. He is a life time ISTE member since 2013. At present, He is working with M.B.E.S. College of Engineering, Ambajogai, Maharastra as an Assistant Professor.