Design of RLE Compression Algorithm using Custom Codesign Technique for WSN Applications

Vilabha S. Patil[1], Shradhha S. Deshpande[2]

[1]Department of E & TC, Rajarambapu Institute of Technology, Rajaramnagar.*
Research scholar, Walchand college of Engineering, Sangli, Maharashtra, India.
[2]Department of Electronics, Walchand college of Engineering, Sangli, Maharashtra, India
[1]Email:vilabha.mane@ritindia.edu
[2]Email: shraddhha.deshpande@walchandsangli.edu

Abstract

Wireless sensor network (WSN) is emerging research area. In WSN applications power consumption is major concern because the sensors have to collect data and transmit to remote sensor nodes for a longer time. Though the data transmission is primary factor in power consumption of sensor nodes, many researchers have focused on decreasing the data transmission using data compression techniques. Thus data compression is required, and many data compression algorithms are used for reducing the data transmission overhead in WSN. The power dissipation of data compression is trading off in contradiction of power saved for transmitting the compressed data. In this paper sensor data collected is compressed using an innovative hardware software codesign approach. The proposed solution addresses the issues of flexibility, scalability using customized acceleration compression algorithm. The design includes NIOSII processor with custom component of Run Length encoding (RLE) which is default choice as data compression method in number of applications. Thus, the paper compares the custom component codesign implementation approach with sequential complete software only implementation. The results represents that performance of proposed approach significantly higher than Software only implementation also leads to power saving.

Keywords: RLE compression, Custom codesign, wireless sensor network, power saving, NIOSII processor

1. INTRODUCTION

WSN is contains number of sensor nodes. Every sensor node made up of different components like sensors, processor, and transreceiver. These sensors nodes collect the surrounding data as per application and transmit it to the receiver node through the several routing algorithms [2]. There are many applications of WSNs like health monitoring, Environment monitoring, agriculture monitoring, indoor surveillance, structural monitoring [4]. Sensor nodes are powered by batteries, thus working of sensor nodes are depends on lifetime of battery. To extend the lifetime of battery, power saving is important criteria [2,4]. One of the major reasons of power consumption is huge data transmission by sensor node. Hence to reduce the amount of data to be sent by sensor nodes by using compression algorithms is the solution which is needed in most of the WSN networks [3, 5]. There are plenty of compression algorithms with two different types lossy and lossless. Lossy algorithms will loose data after decompression. However lossless compression algorithm does not change the data after decompression, it is similar to original data. Therefore the lossless compression techniques are preferred in WSN applications for data compression. The number of compression techniques like Huffman coding, Lumvel ZV, S-LZV, Run length Encoding are emerged over the years. These compression algorithms are computationally demanding, thus processor can dissipate more power (the part of sensor node) due to computations. Due to this the research for hardware implementation of these algorithms has been become more severe during the recent years.

Numbers of ways are available for hardware implementation like microcontroller, microprocessor, FPGA, ASIC. But due to re-configurability and parallel processing major features of Field programmable gate arrays (FPGA) is considered as smart way out for hardware implementation [4].

RLE is the one of the basic lossless technique well suited for WSN applications. Numbers of software implementations for RLE algorithm are developed by researchers [6,7]. In [5] they have developed and evaluated RLE algorithm on MSP430. This software implementation leads to power saving as the implementation is on ultra-low-power microcontroller of Texas Instruments. Another [6] software based implementation has been developed for image compression using Atmega128 processor to evaluate best compression method suitable for video signal

processing. This microcontroller and microprocessor based implementation does not provide required flexibility as well as it will increase the load on processing unit of WSN node in sensor networks. Thus, Engel, A et.al. [8] have developed the reconfigurable architecture "HALOmote" with FPGA based processing. This architecture is utilized for analysis of compression algorithm. This leads to flexible architecture with some power saving. Similarly the hardware acceleration of different compression algorithms is developed in [9]. This hardware only implementation is compared with software only implementation with improved results of energy saving. Energy efficient compression algorithms are introduced in [10] with hardware implementation. It is also proved that compression algorithms lead to reduction in platforms energy dissipation dealing with reducing the duty cycle of radio device. In [11] compression of bit stream generated by FPGA is proposed using RLE, which also helps to improve the bandwidth along with reduction in reconfiguration time. This approach of compression is improving the performance by 15%.

These numbers of researchers have been discovered complete software or complete hardware design and development of RLE compression algorithm. However the hardware software codesign approach with the help of custom instruction has not been designed and developed. This approach leads to parallelism and power saving along with flexibility in design. In this proposed work, RLE compression technique is profiled and synthesized in FPGA leading to parallelism with concurrent execution. Then RLE compression algorithm is designed as custom component using hardware software codesign methodology with NIOSII embedded processor. This proposed design goals to find the significance of RLE compression algorithm implementation as a custom component over the sequential implementation.

2. RUN LENGTH ENCODING (RLE)

It is the basic compression algorithm used to compress the repetition of characters in given data sequence. Most useful aspect of RLE was shown in [12] golomb 1966 . The data is compressed with one pass encode and decode technique using RLE. Another implementation aspect for Fax machine i.e. ITU-T4 was developed in 2010 by Belloch [13]. In this algorithm the idea is that the count of repetition of character will be placed before the character with # sign.

e.g. The string of data like aaaaaaa is encoded in RLE like #a7 i.e. count of repetitions.(# is repetition mark). The RLE compression method is as shown in Fig. 1.

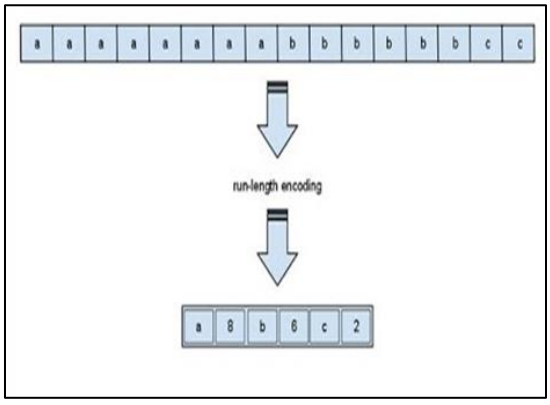

Fig. 1. RLE compression method

Algorithmic Flow:

Fig. 2 illustrates the RLE compression algorithm flow for compression of input 32bit data. It gives the HDL model of RLE function. The flow given in Fig. 2 is implemented in Verilog. This Verilog code is then compiled using Xilinx modelsim simulator. It is synthesized to check its functionality and verify the coding.

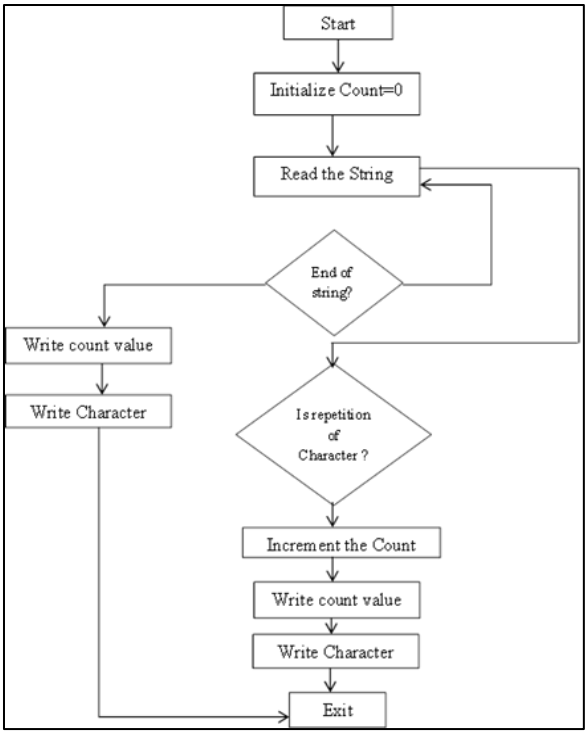

Fig. 2. RLE compression algorithm flow

3.OVERVIEW OF IMPLEMENTATION COMPONENTS

In this proposed work the ALTERA-FPGA, DE2115 board with cyclone IV E device is selected for implementation. This FPGA consists of NIOSII embedded processor with number of features. The considerable features like custom component with hardware-software codesign approach. This approach is used in this paper to improve the performance of compression algorithm. In this design the RLE compression algorithm is implemented in hardware on FPGA as a custom block and co-designed with software using NIOSII soft core processor. Some initial information of design components is provided below:

3.1 Altera FPGA Board

The ALTERA FPGA board DE2115 is chosen for this proposed work which includes following features:

- Cyclone IV E device: It is built using 60nm process. It leads to low power consumption by lowering the core voltage. About 25% of voltage is reduced with the help of this device. This device is optimized for lowest power consumption results in extending the battery life for number of applications.
- Ethernet PHY interfaced with RJ45 connectors with 2 Gigabit
- USB Master and USB slave controller with A/B
- RS-232 transmitter/receiver with 9-pin connector
- Programming modes like AS(active serial) and JTAG are supported with USB Blaster
- Memory modules : SRAM memory component , 64MB SDRAM memory module with Flash memory

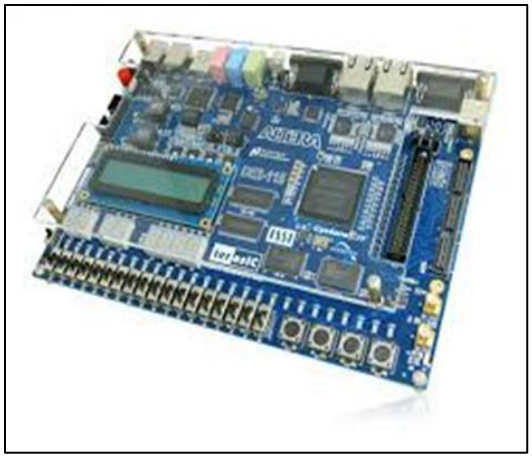

Fig. 3. Cyclone IV Development Board Altera FPGA board

3.2 NIOSII Embedded Processor

It is soft-core processor developed by Intel ALTERA FPGA. It is 16/32bit RISC processor. The NIOSII processors functional capability can be widened by addition of custom components using hardware-software codesign approach. The embedded processor is present with three different versions such as NIOS II/f-fast, NIOS II/s-standard and NIOS II/e-economy. These different versions present with specific performance and specific price. It has RISC architecture with pipelined feature. It has Harward type of architecture with 16-bit instruction bus and 32-bit data bus. About 512 total number of configurable registers are present from which 32 registers are available for general purpose usage.

The total system is integrated on ALTERA DE2 with number of custom peripherals and standard peripherals alongwith NIOSII pocessor. CYCLONE IVE FPGA device used for interfacing of the NIOS II embedded processor in coordination of the peripherals to the DE2115 board. These peripherals and components are connected by using avalon bus by forming the interconnection network.

4. IMPLEMENTATION ON ALTERA FPGA

4.1 The system Design using QSYS

The NIOSII embedded processor is using QSYS (SOPC Builder) as system integration tool for designing hardware system with different components. QSYS is the part of QUARTUSII software used for generation and configuration of NIOSII system design.This integration tool is used to design and develop a system consists of NIOSII processor, memory controller, number of Intellctual Property(IP)-cores, peripherals with custom peripherals.QSYS will automatically generate interconnect logic amongst different components. The system design generated in QSYS is then synthesized with QUARTUSII tool. The placement and routing of generated system design is accomplished using QUARTUSII software tool. Later the system generation has been done for Cyclone IV E FPGA device using system interconnect network.

Thus the system has been designed and developed with QSYS integration tool. It has the following peripherals and components as shown in Fig.4.

- Clock system: The clock source in QSYS tool is selected which will provide clk frequency to all the components in the design with NIOSII processor.
- NIOSII Embedded processor: Two versions of NIOSII processor like NIOSII/e and NIOSII/f are chosen to implement the design to improve the performance of processing unit.
- RLE Compression: This is the custom component, performs RLE compression. The results are stored in RAM memory.
- Memory Component: ON-chip-RAM memory component is chosen for implementation with 48kbyte of size. The program executed from NIOSII IDE. On-chip-memory allows automatic download and execution of program.
- JTAG_UART: The communication and connection between the host_PC and the system design developed using QSYS integration tool is done through serial interface with JTAG_UART.

- RLE Compression custom component: This custom component is designed to increase the performance and reduce the power consumption of data compression technique for WSN applications. It is connected to NIOSII processor as custom component. The custom component takes input data collected from sensor through dataa and datab input ports of custom component. The compressed data will be displayed on NIOS IDE console window.
- Performance counter: This component will be used to measure the performance of compression algorithm bulit as custom component. It will give performance in terms of count of clock cycles and execution time in milliseconds. It will also measure the performance of software only implementation of compression algorithm.
- Input and output ports(GPIOs): Input/output ports are used for purpose of synchronization and testing.

The QSYS system design contains the above mentioned components connected through Avalon bus as shown in figure 4. The data compression is accomplished using RLE custom component in coordination with software program written in NIOS II IDE.

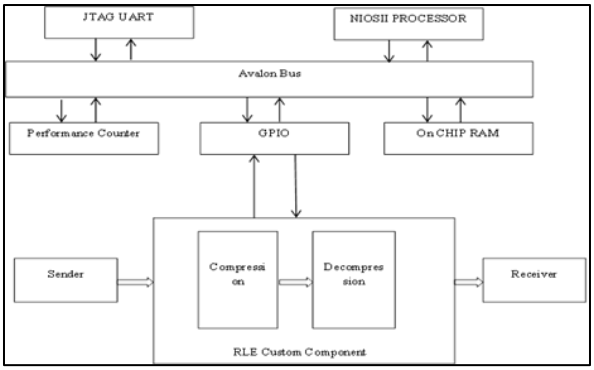

Fig. 4. Connection of RLE in QSYS

Fig.5. Shows the components connected through QSYS builder integration tool.

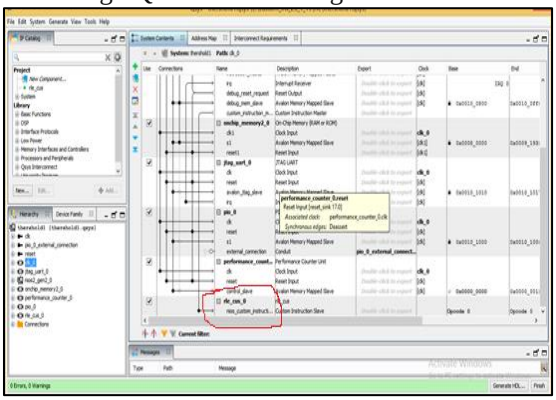

Fig. 5. QSYS system design

As presented in Fig. 5. RLE_cus_0 is a custom component block designed through the hardware software co-design for RLE acceleration through HDL design. This custom block is connected to NIOS II embedded processor using ON-CHIP RAM memory component and JTAG UART.

4.2 Design and Implementation

After generation of system using QSYS SOPC builder tool of QUARTUSII, pin assignments have been done. The design is downloaded to the hardware ALTEA FPGA-DE2115 development board with Cyclone IV E which is connected to PC through JTAG_UART using USB-blaster cable.

Following three steps are done for development and analysis of RLE compression custom component through hardware-software codesign.

4.2.1 RLE implementation in HDL Language

In the section no. 2 the algorithmic flow of RLE compression technique is described. The RLE compression algorithm is developed in Verilog language. This developed Verilog code is compiled with the help of Xilinx ISE. Later the authentication of functionality is done through modelsim simulator. The synthesis and compilation of this design is done through QUATUSII 17.1 version.

4.2.2 RLE implementation using only Software

RLE compression algorithm is developed using C language. The basic system of NIOSII embedded processor is generated in QSYS tool. This basic system is compiled in QuartusII software. With the help of .sopc file the project is created in NIOSII IDE. In this NIOS Eclipse IDE project created with C language is compiled and generated as executable (.exe) file through the GNU compiler. The project created with C language is debugged using build project facility. Then project is downloaded to ALTERA FPGA.

Well ahead the results of software implementation of RLE compression along with performance are displayed on NIOSII console window.

4.2.3 RLE implementation as Custom Component using hardware-software Co-Design approach

RLE compression algorithm is accelerated in hardware using Verilog as custom logic block as mentioned in section 2. This custom logic block is integrated with NIOSII basic system using QSYS SOPC builder tool. There are two main modules of this design as custom component and system.h file generated software function. This software function can be used to access the custom component in the software program written in NIOS II IDE generated through hardware software codesign approach.

The .sopcinfo file – hardware connection is used to generate a project in NIOSII IDE. The basic application program generated in NIOSII IDE is modified with the software function. It is compiled with NIOSII built facility. With the help of run as NIOSII hardware the project is executed and so that the output will be displayed on console window as given in Fig. 6 i.e. the compressed form of input data using RLE custom component with the performance factors such as clock cycles count and execution time required for compression.

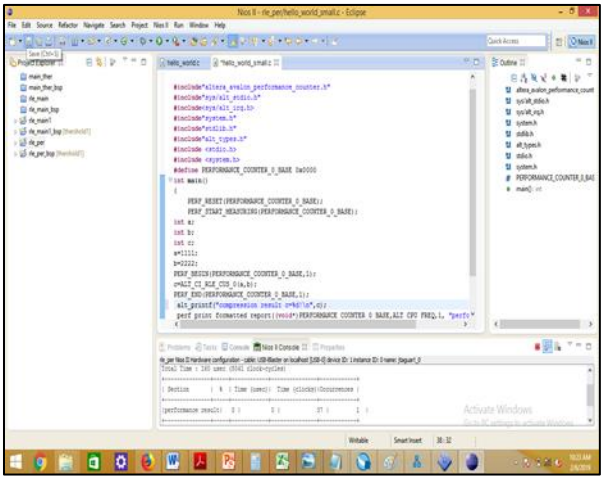

Fig. 6. NIOSII console window

5. RESULT AND DISCUSSIONS

The RLE compression software program generated in NIOS IDE with and without custom instruction is compiled in QUARTUSII environment. The result in terms of performance parameters like count of clock cycle, time of execution and power consumption for the respective two purposes is compared. The custom component approach and software approach have been tested for two NIOSII versions i.e. for NIOSII E and NIOSII F. The results for count of clock cycle and time of execution for both approaches with respective versions are as shown in below Table 1:

Table 1. Count of clock cycles and Execution time

Item	Without custom instruction (software Implementation)		With Custom Instruction Implementation	
	NIOSII E	NIOSII F	NIOSII E	NIOSII F
Clock Cycles	17285	3669	37	14
Time of Execution (µSec)	345	73	0.74	0.28

The results in Table 1. shows that the performance parameters i.e. count of clock cycles and execution time of RLE compression technique implementation with software program (without custom component) are more in comparison to the implementation of RLE compression technique with custom component approach. The custom instruction implementation requires 90 % less clock cycles as compared to software only implementation of algorithm (without custom instruction). Thus codesign approach of custom instruction leads to significant reduction in clock cycles and execution time.

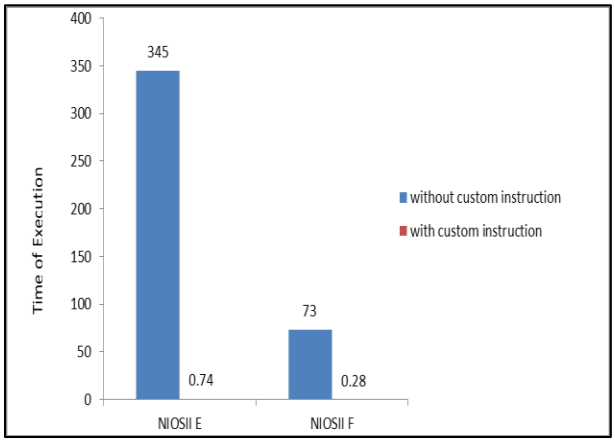

Fig. 7. Comparison of execution time

Fig. 7 shows the graphical comparison of the execution time of RLE compression algorithm for both versions of NIOSII. Performance of processor can be evaluated with the help of clock cycle count as a performance metric. Thus the enhancement in efficiency due to this proposed implementation in terms of clock cycle is evaluated as:

- $$Performance = \frac{\text{NIOS II Processor frequency (50 MHz)}}{\text{Count of clock cycles essential for given algorithm implementation}} \quad (1)$$

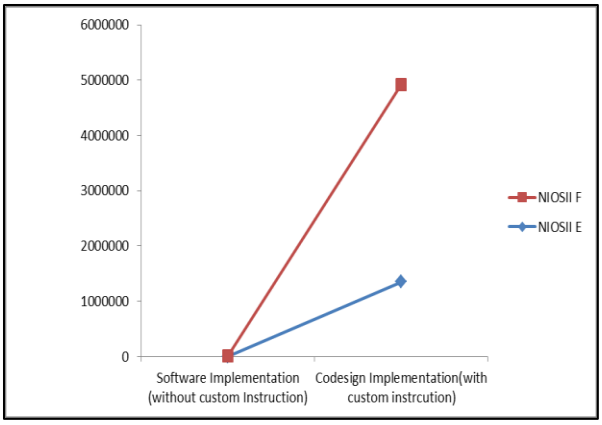

Fig. 8. Comparison of performance

From the Fig. 8, it is identified that hardware software codesign implementation with custom instruction is much improved performance.

This implementation also leads to power saving. The power dissipation results of this proposed implementation are considerably reduced as compared with sequential software implementation (without custom instruction) as shown in Fig. 9.

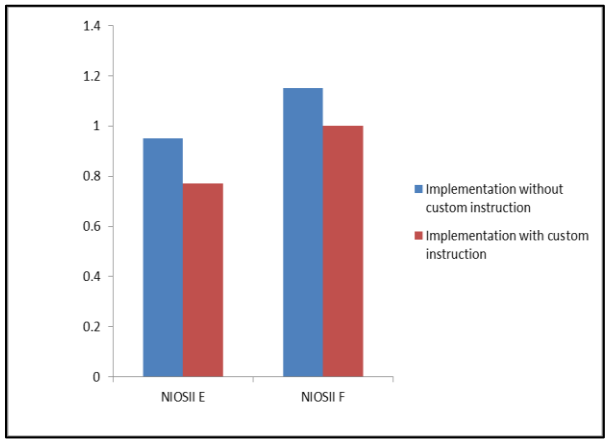

Fig. 9. Comparison of Power Consumption

6. CONCLUSION

Run Length Encoding (RLE) compression algorithm is needed in WSN application for data compression. It is implemented using custom codesign approach as a custom component (instruction). This design leads to optimized approach for design of RLE algorithm which will reduce the overhead on processing unit due to its parallel hardware software codesign approach. RLE is designed and implemented on CYCLONE IV E based embedded processor NIOSII. This work gives speed improvement by 90% and performance improvement by 95%. Additionally it leads to 10% power saving with respective implementation. Thus, the proposed design helps for flexible architecture with significant performance improvement with less execution cycles.

REFERENCES

[1]Marcelloni, Francesco, and Massimo V.A, simple algorithm for data compression in wireless sensor networks. IEEE communications letters 12.6 2008, 411-413.

[2]Anastasi, Giuseppe, et al. How to prolong the lifetime of wireless sensor networks, Mobile Ad Hoc and Pervasive Communications, 2006: pp-1-26.

[3] Fasolo E, Rossi M, Widmer J, and Zorzi M, In-network aggregation techniques for wireless sensor networks: a survey. IEEE Trans. Wireless Commun., vol. 14, no. 2Apr. 2007:pp. 70–87

[4] Patil, V. S., Mane Y. B., Deshpande, S. FPGA Based Power Saving Technique for Sensor Node in Wireless Sensor Network (WSN), In Computational Intelligence in Sensor Networks, pp. 385-404, Springer, Berlin, Heidelberg, 2019.

[5] Bin. Zheng, et al. A compression algorithm in wireless sensor networks of bearing monitoring. Journal of Physics: Conference Series, Vol. 305, No. 1, IOP Publishing, 2011.

[6] Capo-Chichi, Eugène Pamba, Hervé Guyennet, and Jean-Michel Friedt, K-RLE: a new data compression algorithm for wireless sensor network. third International Conference on Sensor Technologies and Applications. IEEE, 2009.

[7] Slabý R, Radim H, and Zdeněk M, Compression methods for image processing implementation into the low capacity devices. 2013.

[8] Engel Andreas, Björn Liebig, and Andreas Koch. Feasibility analysis of reconfigurable computing in low-power wireless sensor application. International Symposium on Applied Reconfigurable Computing. Springer, Berlin, Heidelberg, 2011.

[9] Engel, Andreas, and Andreas Koch. Hardware-accelerated data compression in low-power wireless sensor networks .International Symposium on Applied Reconfigurable Computing. Springer,Cham, 2014.

[10] Reinhardt, Andreas, et al. "On the energy efficiency of lossless data compression in wireless sensor networks", IEEE 34th Conference on Local Computer Networks. IEEE, 2009.

[11] Karthick, S., S. Dhivyapriya, and T. V. P. Sundararajan. Compression of FPGA bit streams using effective run length encoding techniques and its performance estimation.

[12] S.W.Golomb,Run-Length Encodings. IEEE transactions on Information Theory, 1966,12(3) , pp-399-401.

[13]Guy E. Blelloch, Introduction to Data Compression. Computer Science Department, Carnegie Mellon University, September ,2010.