International Journal of Mechanical and Production Engineering Research and Development (IJMPERD)
ISSN(P): 2249–6890; ISSN(E): 2249–8001
Vol. 10, Issue 3, Jun 2020, 12787–12792
© TJPRC Pvt. Ltd.

AN EFFICIENT HARDWARE IMPLEMENTATION OF DATA COMPRESSION USING MODIFIED GOLOMB RICE CODING ARCHITECTURE

M. PRADEEP[1], K.V. SUBRAHMANYAM[2], P. KAMALAKAR[3] & P. RAJESH[4]

[1] *Department of ECE, Shri Vishnu Engg, College for Women, Bhimavaram, Andhrapradesh, India*

[2] *RISE Krishna Sai Gandhi Group of Institutions, Ongole, Andhrapradesh, India*

[3] *Department of EEE, MallaReddy Engineering College(A),Hyderabad, Telangana, India*

[4] *Department of ECE, Rathinam Technical Campus, Coimbatore, India*

ABSTRACT

Golomb coding is a loss-less data compression method using a family of data compression codes. Rice codes are a class of Golomb codes in which the divisor is valid only when it can be expressed as power of two. This work aims to increase compression by further encoding the unary part of the Golomb Rice (GR) code to decrease the numberof bits used. The algorithm was developed and coded in Verilog Hardware Description Language (VHDL) and simulated using Modelsim. The modifications carried out and reduction in bits used for a linearly distributed data set. Worst-case delays have been reduced acceptable level. Area reduction varies for different methods. An approximate 4% to 7% reduction in power consumption is observed. Modified Golomb Rice coding is used to compress multiple data types, which should ideally have a geometrical distribution.

KEYWORDS: Golomb Rice Algorithm, Data Compression, Image Processing and VHDL

Original Article

Received: Jun 08, 2020; **Accepted:** Jun 28, 2020; **Published:** Sep 14, 2020; **Paper Id.:** IJMPERDJUN20201217

INTRODUCTION

S.W.Golombin the 1960s introduced the Golomb coding method. This algorithm is one of the lossless data compression methods and is using a family of data compression codes [1]. Golomb coding is highly suitable for situations in which the occurrence of small values in the input stream is significantly more likely than large values. In general, the images are classified in to two types First one is a Gray Scale image and second one a Colour Image. In analysis, compared with gray scale image, the colour image needs more attention and observation. In Image processing to reduce the memory size of the image and bandwidth, a high-performance lossless compression and decompression techniques is used [2-3]. Such cases, two algorithms are mainly used namely, Raster file and Golomb Rice Algorithm.

Embedded system applications and performance improvements the compressed memory schemes are used. Operating system-based and Discrete Cosine Transform based memory compressions are mainly focused areas in embedded system. Hadamard transform and Golomb-Rice coding are proposed for many researchers in hardware architecture-based Image compression [4]. Some demerits are in both Hadamard transform and Golomb-Rice encoding because of losses in data. So, overcome this difficulties and maintain the lossless data compression and decompression can be used to achieve low bit rate and improve imagequality [5].

Golomb Rice Coding

The sensor data and reduced data loss one of the efficient algorithms is Rice Golomb coding [5-7].

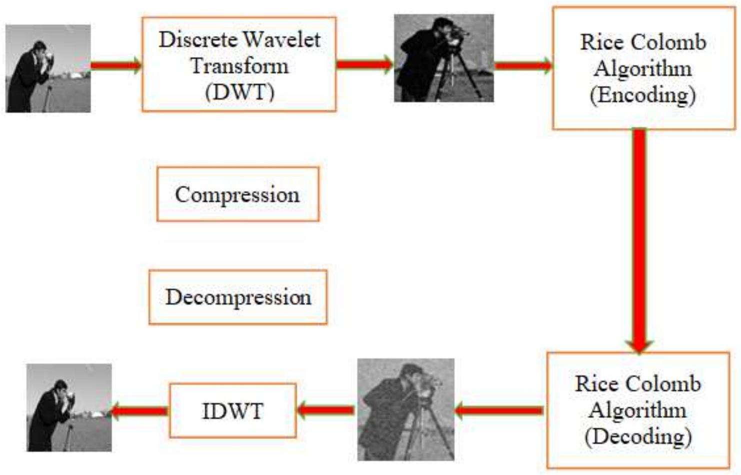
Discrete Wavelet Transform (DWT)
Rice Colomb Algorithm (Encoding)
Compression
Decompression
Rice Colomb Algorithm (Decoding)
IDWT

Figure 1: Block diagram of Image compression and Decompression.

In Figure 1, shows the discrete wavelet transform based compressed input image. The output of the 2-Dimensional discrete wavelet transform is in the form of sub-windows of m x n arrays. Then the image is converted into differential data by using raster file conversion. By using Golomb-Rice algorithm differential data has been encoded to produce length and code word of the input image. Based on the length and code word obtained the encoded image has been Decoded using Golomb-Rice algorithm. Thedecoded image has been decompressed using the inverse 2-D discrete wavelet transform. The output produced is similar to that of the input image.

OBJECTIVES

- To analyze and identify whether the proposed technique is a valid methods for image compression anddecompression.
- To compare the proposed method to exitingmethods.
- To prove, the proposed technique is efficientone.

METHODOLOGY

A simple procedure is adopted here for the easy implementation of Golomb, and it gives the pre ‘x’codes for the suboptimal. By a single parameter, the proposed method codes are distinguished from each other, in order to achieve dynamic update and accomplished for parameters estimations. Here, the methods are explained step by step procedure. From, countable alphabet the symbols are encoded by Golomb codes. All the encoded symbols are arranged in descending probability order. The Non negative integers beginning with ‘0’and the total number of encode integer is assigned value ‘n’. The ‘g’ is Golomb parameter. The ‘reset’ (rst) compute ‘hn=gc’ and output this integer using a unary code. Then we compute n mod m and output this value using a binary code, adjusted as described above so that we sometimes use ‘gc’bits and sometimes ‘ga’bits. Golomb coding and Rice coding are most probably same algorithm. Hence to encode integer n using the Rice code with parameter ‘z’ and ‘rst’ compute ‘hn=2kc’ and output this integer using a unarycode. Golomb-Rice

Encoder main task – as described before – is to generate code and length values. At each clock cycle, it receives the residual values of four pixels of one specific subtle (a1, a2, a3, and a4) of one component, and generates corresponding “code”and “header” values for each pixel. The Encoder block consists of three sub-blocks shown in Figure 2.

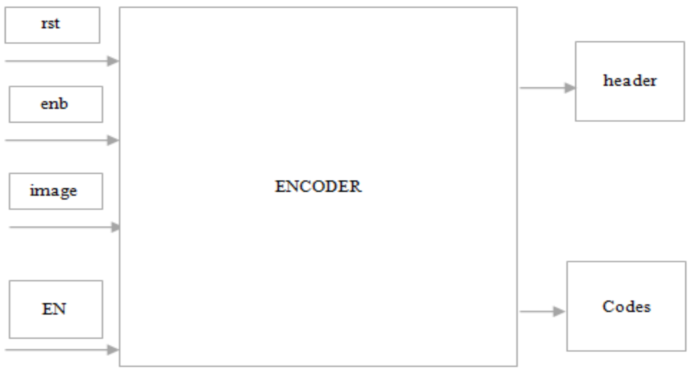

Figure 2: Encoder Block.

RESULTS AND DISCUSSIONS

The compressor and decompressor data path blocks are synthesized with the target clock frequency of 220MHz. The overall block size of the compressor is 9.20kgates. This size includes data path blocks; input-output data registers and address generators. 59.2 % of this block is combinational, and the rest is sequential logic size. In order to verify the functionality of the design, a verification framework has been designed. This framework consists of oneRAM blocks, a counter to generate the address for the memory, and a control block which is a finite state machine. There are three modes of operation during the functional verification. The input image to be compressed has already been converted to binary representation in MATLAB and stored in a text file as input file to the framework. This file has 32 columns which is equal to the 32-bit memory word-length. The number of lines in the file depends on the image size and therefore the memory size has to be adjusted accordingly. In the first mode, content of the input file is stored in the source memory, RAM. It takes ‘n’ clock cycle to perform this data transfer where ‘n’is the number of lines in the input file. That is when the FSM issues “Execution” signal and the operation enters the next mode where the compressor/decompressor starts performing its task and sending the result to the destination memory, RAM. When it issues “Execution Complete” signal, FSM changes to the final mode where the content of RAM is written to the output text file. The functional verification blocks are shown in Figure3.

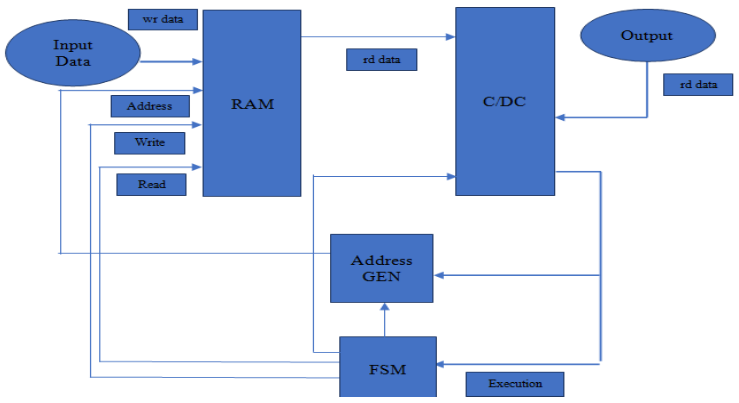

Figure 3: Functional Verification Frame.

The overall block includes 618 registers. The sub-block sizes inside the datapath are also of interest. Table 1 shows the hierarchical area distribution of the whole block.

Table 1: The Hierarchical Area Distribution Compressor

S.No.	Block Name	Area (kgates)
1.	Compressor	9.20
2.	Color Transform 1	0.23
3.	Color Transform 2	0.21
4.	Golomb- Rice Encoder	2.34
5.	Encoder 1	0.37
6.	Encoder 2	0.36
7.	Encoder 3	0.37
8.	Encoder 4	0.37
9.	GR Control	0.48
10.	GR Parameter Estimation	0.36
11.	Input Preparation	0.67
12.	Address generator 1	0.59
13.	Address generator 2	0.45
14.	Encoder register file control	1.87
15.	Prediction Register File control 1	0.65
16.	Prediction Register File control 2	0.66
17.	Prediction Register File control 3	0.72
18.	Prediction Register File control 4	0.73
19.	Predictor 1 -4	0.35

The overall block size of the decompressor is 9.23kgates. This size includes data path blocks; input-output data registers and address generators. 59.1 % of this block is combinational, and the rest is sequential logic size. The overall block includes 498 registers. The sub-block sizes inside the data path are also of interest. Table 2 shows the hierarchical area distribution of the whole block. An important result that can be extracted from synthesis is that functional blocks constitute relatively small portion of the overall cost. The more costly operations are the ones that are related with the traversal of the image. Also due to high throughput requirement (one pixel/clock in our case) the pipeline registers and temporary storage registers constitute a significant portion of overall size. In that sense, it would not be wrong to claim that fast implementations of such simpler algorithms are control dominated in terms of hardware cost. Another result is that the Ctrl block, which is added to improve compression ratio, only takes 470 gates to implement which corresponds to 3.9% of the overall compressor data path.

Table 2: The Hierarchical Area Distribution - Decompressor

S.No.	Block Name	Area (kgates)
1.	Decompressor	9.23
2.	Color Transform	0.29
3.	Golomb- Rice Encoder	0.27
4.	Output preparation	1.46
5.	Reverse Color Transform	0.18
6.	Address Generator 1	0.68
7.	Address Generator 2	0.46
8.	Construction register file control 1	0.51
9.	Construction register file control 2	0.58
10.	Construction register file control 3	0.58
11.	Construction register file control 4	0.58
12.	Constructor 1- 4	0.36
13.	Decoder register file control	1.61

CONCLUSIONS

The proposed work carried out in this paper investigated Modified Golomb Rice algorithms from hardware implementation point of view to be used in high throughput hardware. One such algorithm is aimed to be implemented in hardware in order to validate early cost estimations and gain more insight into problematic parts of the algorithms in terms of hardware implementation. A possible hardware realization for functional blocks of compression and decompression is proposed in this paper. Then, the work continued with introducing a compression algorithm based on proposed algorithm for in order to get a better compression performance. This proposed algorithm is chosen for hardware implementation in VHDL. In future various parameters will be considered for evaluation of the proposed algorithm and will be compared with different methods for optimum solution.

REFERENCES

1. *S. Kim and N. I. Cho, "Lossless Compression of Color Filter Array Images by Hierarchical Prediction and Context Modeling," IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 6, pp. 1040-1046,2014.*
2. *S.Kalaivani and C.Tharini, "Analysis and implementation of novel Rice Golomb coding algorithm for wireless sensor networks" ComputerCommunications,Volume 150,15January 2020, Pages 463-471, https://doi.org/10.1016/j.comcom.2019.11.046.*
3. *Basha, S. M., &Jinaga, B. C. (2013). An Optimum Novel Technique Based on Golomb-Rice Coding for Lossless Image Compression of Digital Images. International Journal of Signal Processing, Image Processing and Pattern Recognition, 6(5), 291-304.*
4. *S. L. Chen, Y. R. Chen, T. L. Lin, Z. Y. Liu. "A cost-efficient lossless compression color filter array images VLSI design for wireless capsule endoscopy." Journal of Medical Imaging and Health Informatics. Vol. 5, no. 2, pp. 378-84,2015.*
5. *Golomb, S.W., 1966.Run-length codings. IEEE Trans. Inform. Theory,12(7): 399–401.*
6. *K. Somasundaram and P. Sumithra, "A Novel method to compress still images using Golomb code (GC) in JPEG2000," International Journal of Complete Science and Network Security, vol. 10, no. 8, pp. 182–186,2010.*
7. *P. Eben Sophia and J. Anitha," A systematic review on advances and perspectives of image compression in telemedicine", International Journal of Advanced Intelligence Paradigms, 7(2), pp.136–155.*