International Journal of Emerging Trends in Engineering and Basic Sciences (IJEEBS)
ISSN (Online) 2349-6967
Volume 2, Issue 3(May-June 2015), PP 18-23

Lossless Data Compression Using Golumb Codes

Mr. Suhas D. Kakde[1] , Mr. Jayantkumar Dorave[2], Mr. Raman Bondre[3]
[123]*(Deptt. Of E&TC Engg., Priyadarshini J L College of Engineering, Nagpur/RTMNU Nagpur, India.)*

Abstract:- This paper describes the detailed study and analysis of Golumb codes used for the test vector compression in VLSI testing. The Golumb Codes are widely used for lossless Data compression due to its simpler encoding & Decoding methods. Furthermore, a comparative study of various compression techniques is also presented in this paper. This paper also give out an idea about the design of Golumb Encoder & Decoder using VHDL for the test vectors thus achieving a good height of compression ratio. The developed algorithm can be verified using the Modelsim 6.3f , Xilinx 9.2i and ALTERA Quartus II software.

Keywords:- Golumb, Compression, lossless, EGC, MEGC

I. INTRODUCTION

The rapid widespread growth of digital technologies such as digital television, internet access, huge amount of data storage and video calls have increased the demand for high storage data and transmission capacity in order to fit the growing needs[1]. Data compression involves encoding information using fewer bits than the original representation. Compression of data in large-sized files reduces storage space and transmission time in a network and hence reduces the cost of storage and transmission. Most of the files has lot of redundancy which can be removed using Compression.

Compression can be achieved through a lossy or lossless mechanism. The selection of compression method depends on the application. Lossy compression of a tolerable limit can be used for video transmission, where the loss of data in the reconstruction cannot be easily noticed by human visual system. However, compression of data of crucial information such as the database of a bank needs to be lossless. There have been extensive research efforts in this field since the last 50 years.

Golomb coding is one of the lossless data compression techniques. It is competent of compressing larger sized data into a smaller sized data while still allowing the original data to be reconstructed back after decompression [1]. Due to its lower design complexity and computational load, Golumb codes are more preferred over other lossless data compression codes.

Exp-Golomb VLC for entropy coding in H.264, which is mostly used for video coding standard is the most efficient application of golumb code . Golumb codes are also used for colour image [4], [6] and audio compression. Other application of golumb codes is for embedded cores in a system-on-a-chip (SOC) [7]. The key features of Golomb coding for test data include very elevated compression, analytically predictable compression results, and a stumpy cost and scalable on-chip decoder.

In this paper, a detail study of Golomb coding algorithms for test vector compression and decompression is presented. In order to have simplicity in development and testing, the Golomb coding parameter m is set to 4 [2]. The goal of this work was to increase the compression ratio as high as possible without any loss in the original data. The remainder of the paper is organised as follows. Section 1 presents the details of Golomb Coding and the basic compression and decompression method. Section 2 presents the review of past work carried out on golumb codes. Section 3 presents the future scope in the field of Golumb Codes. Lastly, Section 4 concludes the paper.

II. REVIEW OF GOLUMB CODES

Golomb coding is a lossless data compression method using a family of data compression codes invented by Solomon W. Golomb in the 1960s. Alphabets following a geometric distribution will have a Golomb code as an optimal prefix code, making Golomb coding highly suitable for situations in which the occurrence of small values in the input stream is significantly more likely than large values.

A. Golumb Encoder and Decoder Algorithms

The details regarding Golomb Coding basic background information is described. In Golomb Coding, the group size, m, defines the code structure. Thus, choosing the m parameter decides variable length code structure which will have direct impact on the compression efficiency [2]. After finalization of parameter m, a table which maps the runs of zeros until the code is ended with a one is created. A Run length of multiples of m are grouped into Ak and given the same prefix, which is (k – 1) number of one's followed by a zero, which can also termed as quotient and can be represented in the form of unary codes. A tail is given for each member of the group, which is the binary representation of log2m bits. The other term for tail is Remainder of the division of run length by m. The codeword is then produced by combining the prefix and the tail. An example of the analytically encoding the data stream using Golumb Encoder is given in table 2.

Group	Run length	Group prefix (Quotient)	Tail (Remainder)	Codeword (Prefix + tail)
A1	0	0	00	000
	1		01	001
	2		10	010
	3		11	011
A2	4	10	00	1000
	5		01	1001
	6		10	1010
	7		11	1011
A3	8	110	00	11000
	9		01	11001
	10		10	11010
	11		11	11011

Table I. Golomb Encoding example with parameter m = 4

Data Set	00100010000100000010000001000101						
Subset	001	0001	00001	0000001	000001	0001	01
Encoded Output	010	1000	1001	1010	1001	011	001

Table II. Golomb Encoding example with parameter m = 4

The golomb Encoder and Decoder models can be well explained with the help of flowcharts given in figure 1 and figure 2 [2]. The design of the algorithm is made with the assumption that the input data string will be terminated with a "1". This will not always be the case because the input data string may also be terminated with

a "0".To overcome this problem, a modified version of Golumb Compression and decompression methods are used. In order to avoid this problem, the algorithm must be capable of detecting the end of data and if the last bit is a '0' then additional '1' must be added during the encoding process and at the time of decompressing the encoded data,this extra appended 1 should be removed by the decoder.

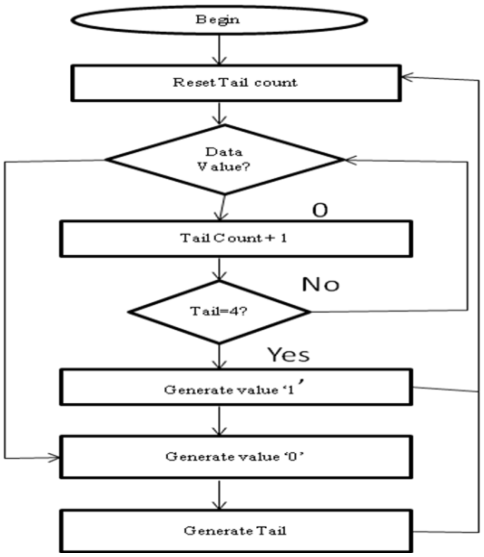

Figure 1. Golomb Encoder Flowchart with parameter m =4

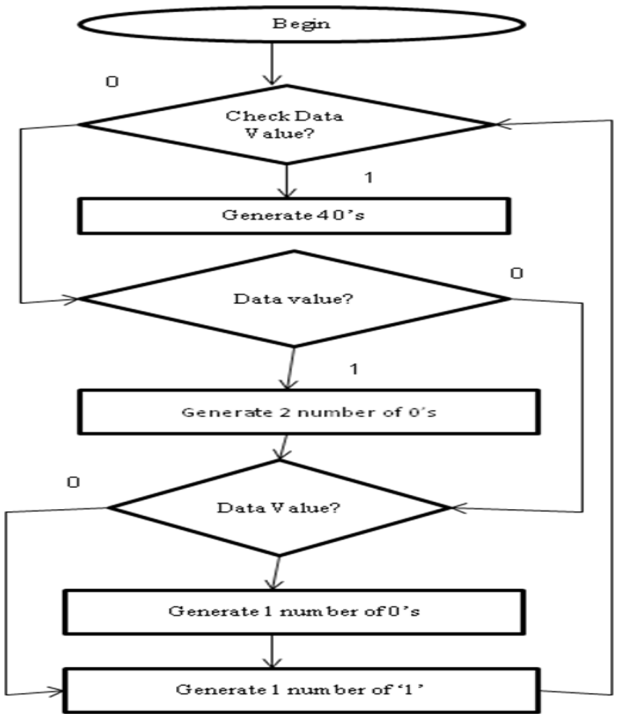

Figure 2. Golomb Decoder Flowchart with parameter m = 4

B. A lossless color image compression and decompression hardware architecture

A lossless color image compression and decompression hardware architecture reduces the memory requirement and bandwidth of the system to a larger extent. The architecture consists of differential-differential pulse code modulation (DDPCM) and Golomb-Rice coding. The original image frame is organized as m by n sub-window arrays, to which Differential differential Pulse code modulation (DDPCM) is applied to produce one seed and m×n−1 pieces of differential data [4]. Then using the Golomb algorithm, the differential data are encoded to produce lossless compressed code. This gives the high compression rate and throughput at the output for real time lossless Compression and decompression operations for small area. The performance comparison between the DDPCM + Golumb Encoding scheme and some other encoder algorithms like JPEG-LS, FELICS & arithmetic codes is given in table 3.

Algorithm	Arithmetic Coding	FELICS	JPEG-LS	DDPCM + Golumb Rice
Technology	Xillinx vertex 4	0.13um	0.18um	0.15um
Operating frequency	123	273	40	170
Throughput (MB/s)	123	546	9.9	2720
Parallelism	1	2	1	16
Compression ratio	1.76	235	NA	1.52
Area (GEs)	NA	12.97k	17.6k	16k
Power Efficiency	NA	16	1.38	607.14

Table 3. Encoder Comparison scheme

The Receiver side consists of three steps of unpacking, Golomb-decoding, and inverse DDPCM. The unpacking hardware module decides whether the input is compressed or not according to the value of the V bit stored in the V-buffer. If the V bit stored in V buffer is 1, then the data is directly forwarded to the processor without any processing. Or else, Golomb-Rice decoding and inverse DDPCM operations are performed to reconstruct the original 4 × 4 sub-window image data. The circuit at the output side is divided into two pipeline stages in which the first stage is Golumb decoding and the differential output of decoder is provided as input to second stage which is inverse DDPCM. Table 4 gives the performance comparison scheme between Golumb Decoder + DDPCM and Lossless JPEG decoder.

Algorithm	Technology	Operating frequency	Throughput (MB/s)	Logic Area (GEs)
DDPCM+ Golumb decoderRice [4]	0.15um	170	270	16.46k
Lossless JPEG decoder [11]	0.18um	200	500	39k

Table 4. Decoder Performance comparison scheme

C. Extended Golomb Code for Integer Representation

Two methods to represent nonnegative integers based on the principle of Golomb code is described in this section. In both methods, the given integer is successively divided with a divisor, the quotient and the remainders are then used to represent the value of non-negative integer. First method is best suited for representing short integers. Another method is best suited for representing both short and long integers.

Extended Golumb codes(EGC) is a method to code the given nonnegative Integer n[5]. In this method, a divisor d is selected and the integer n to be coded is divided successively m times by d until the quotient q becomes zero. In each division, the remainder value is retained. The integer is then coded by coding m and the m remainders as
code(m)|code(rm,rm-1,…….r1)

N	d=2	d=3	d=4
1	1	10	10
2	010	11	110
3	011	0100	111
4	00100	01010	01000
5	00101	01011	01001

Table 5. Extended Golumb Code Example

The second method is Modified Extended golumb code (MEGC) [5], in which the successive division of the given integer n by the divisor d was done until the quotient q becomes zero. In Modified Extended golumb code (MEGC), an integer n is divided by a divisor d successively until the quotient q becomes either 0 or 1 unlike the previous Extended Golumb Codes where successive divisions were made until the quotient becomes zero. The last remainder rm is coded in log2 (d-2) bits when m>=2 and q=0 , in log2(d-1) bits when m=1 and q=0 and in log2(d) bits for all other cases. The advantage of Modified Extended golumb code (MEGC) is that, if the division is stopped when q reaches the value 1 in that case, large integers will get benefited in its length of code. In MEGC, short integers take more number of bits and large integers take less number of bits than EGC. Table 5 and Table 6 gives the example of Extended Golumb Encoding and Modified Extended Golumb Encoding with different values of divisor.

N	d=2	d=3	d=4
1	0	010	010
2	110	011	0110
3	111	110	0111
4	10100	1110	1100
5	10101	1111	1101

Table 6. Modified Extended Golumb Code Example

III. PROJECTED SCOPE AND METHODOLOGY

The development of Golomb coding algorithms for data compression and decompression and their implementation in Field Programmable Gate Array (FPGA) is studied successfully. In Simple Golumb codes, the data is divided into subsets which maps the runs of zeros until the code is ended with a one[2]. In future work of Golumb coding,a modified Golumb Coder is designed to serve as a good application for the test vector Compression. In that case, the input bit stream is divided into two subsets in which first subset will contain the test vectors which maps the runs of zeros until the code is ended with a one and the second subset will contain the vectors which maps the runs of one's until the code is ended with a zero. For first subset, the data is divided

by the divisor m and in the second subset, the data is divided by the divisor n. For lower error probability, the group size m and n for both the subsets is kept equal. The optimum value of m and n is set to 4. For verification, simulation and hardware realization Golomb Coding compression and decompression algorithms can be implemented on the Field Programmable Gate Array (FPGA). The Modified Golumb Coder will also serve as a good application for compressing an image.

IV. CONCLUSION

In this paper, algorithms of Golumb encoder and decoder for test vector compression has been studied. A proportional study is also proposed between Golumb Encoder and other data compression techniques. The successful working of these algorithms can be proved by comparing the results generated using simulation of the Golomb Encoder with the expected analytical results which should be identical. The use of encoded data as the input data for Golomb Decoder managed to generate back the original data can prove the success of Golomb Decoder System. ensure a high-quality product, diagrams and lettering MUST be either computer-drafted or drawn using India ink.

REFERENCES

This heading is not assigned a number.

A reference list **MUST** be included using the following information as a guide. Only *cited* text references are included. Each reference is referred to in the text by a number enclosed in a square bracket (i.e., [3]). References **must be numbered and ordered according to where they are first mentioned in the paper,** NOT alphabetically.

[1] S. W. Golomb, "Run Length Encodings," *IEEE Transactions on Information Theory*,*vol.12*, pp. 399-401, 1966.
[2] G. H. H'ng, M. F. M. Salleh and Z. A. Halim," Golomb Coding Implementation in FPGA", School of Electricaland Electronic Engineering, Universiti Sains Malaysia,Seri Ampangan,14300 Nibon Tebal, Pulau Pinag, Malaysia. VOL. 10, NO. 2, 2008, 36-40.
[3] Walter D. Leon-Salas, Sina Balkir, Khalid Sayood, and Michael W. Hoffman, "An Analog-to-Digital Converter with Golomb–Rice Output Codes" *IEEE transactions on circuits and systems—ii: express briefs, vol. 53, no. 4. APRIL 2006.*
[4] Hong-Sik Kim, Joohong Lee, Hyunjin Kim, Sungh Kang, and Woo Chan Park, A Lossless Color Image Compression Architecture using a Parallel Golomb- Rice Hardware CODEC, *IEEE transactions on circuits and systems for video Technology, vol. 21, no. 11, November 2011.*
[5] K. Somasundaram and S. Domnic, "Extended Golomb Code for Integer Representation", *IEEE transactions on multimedia, VOL. 9, NO. 2, FEBRUARY 2007.*
[6] Chin-Chen Chang, "An Enhancement of JPEG Still Image Compression with Adaptive Linear Regression and Golomb-Rice coding, 2009 *Ninth International Conference on Hybrid Intelligent Syst*ems.
[7] Anshuman Chandra, Student Member, IEEE, and Krishnendu Chakrabarty, Senior Member, IEE " System-on-a-Chip Test-Data Compression and Decompression Architectures Based on Golomb Codes",*IEEE transactions on computer aided design of integrated circuits and systems, VOL. 20, NO. 3, MARCH 2001*.
[8] Henrique S. Malvar, "Lossless and Near-Lossless Audio Compression Using Integer- Reversible Modulated Lapped Transforms, 2007 *Data Compression Conference*.
[9] X. Chen, N. Canagarajah, J. L. Nunez-Yanez, and R.Vitulli, "Hardware architecture for lossless image compression based on context-based modeling and arithmetic coding," in Proc. *IEEE Int. SoC Conf., Sep. 2007, pp. 251–254.*
[10] T.-H. Tsai, Y.-H. Lee, and Y.-Y. Lee, "Design and analysis of high throughput lossless image compression engine using VLSI-oriented FELICS algorithm," *IEEE Trans. Very Large Scale Integr. Syst., vol.18, no. 1, pp. 39–52, Jan. 2010.*