P-3470

IMAGE COMPRESSION ENGINE USING VLSI-ORIENTED FELICS ALGORITHM

By

P.GEETHA

Reg. No. 1020106005

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University, Coimbatore)

COIMBATORE - 641049

A MINI PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

of

MASTER OF ENGINEERING

IN

APPLIED ELECTRONICS

MAY 2011

BONAFIDE CERTIFICATE

Certified that this project report entitled **"IMAGE COMPRESSION ENGINE USING VLSI-ORIENTED FELICS ALGORITHM"** is the bonafide work of **Ms.P.GEETHA [Reg.no.1020106005]** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible]

Project Guide

Prof.S.Govindaraju

[illegible]

for **Head of the Department**

Dr.Rajeswari Mariappan

The candidate with university Register no. 1020106005 is examined by us in the project viva-voce examination held on ...23.05.11........

[illegible]

Internal Examiner

[illegible] 23/5

External Examiner

ACKNOWLEDGEMENT

A project of this nature needs co-operation and support from many for successful completion. In this regards, I am fortunate to express my heartfelt thanks to Chairman **Padmabhusan Arutselvar Dr.N.Mahalingam B.Sc.,F.I.E.,** and Co-Chairman **Dr.B.K.Krishnaraj Vanavarayar B.Com,B.L.,** for providing necessary facilities throughout the course.

I would like to express my thanks and appreciation to the many people who have contributed to the successful completion of this project. First I thank, **Dr.J.Shanmugam, Ph.D,** Director, for providing me an opportunity to carry out this project work.

I would like to thank **Dr.S.Ramachandran, Ph.D,** Principal, who gave her continual support and opportunity for completing the project work successfully.

I would like to thank **Dr.Rajeswari Mariappan, Ph.D,** Prof of Head, Department of Electronics and Communication Engineering, who gave her continual support for us throughout the course of study.

I would like to thank **Prof.S.Govindaraju, (Ph.D),** Professor, Project guide for his technical guidance, constructive criticism and many valuable suggestions provided throughout the project work.

My heartfelt thanks to **Ms.R.Hemalatha, M.E.,** Associate Professor, Project coordinator, for her contribution and innovative ideas at various stages of the project to successfully complete this work.

I express my sincere gratitude to my family members, friends and to all my staff members of Electronics and Communication Engineering Department for their support throughout the course of my project.

ABSTRACT

VLSI-oriented fast, efficient, lossless image compression system (FELICS) algorithm, which consists of Simplified adjusted binary code and Golomb--Rice code with storage-less k parameter selection. The aim of this project is proposed to provide the lossless compression method for high-throughput applications. The simplified adjusted binary code reduces the number of arithmetic operation and improves processing speed. According to theoretical analysis, the storage-less k parameter selection applies a fixed value in Golomb--Rice code to remove data dependency and extra storage for cumulating table. Based on VLSI-oriented FELICS algorithm, the proposed hardware architecture can be extended to multilevel parallelism for advanced high-definition (HD) display specifications and it also enhances regular data flow, and two-level parallelism with four-stage pipelining.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Due to the great innovation of information technology, the stringent requirement of data capacity is drastically increased in communication. The data compression technique is extensively applied to offer acceptable solution for this scenario, and it can be classified into two categories: lossy and lossless. For lossy compression technique, many standards have been intensively developed such as JPEG and JPEG 2000 for still image, and MPEG-4 and H.264 for multimedia communications and high-end video applications, respectively.

Lossless compression can remove redundant information and guarantee that the reconstructed procedure is without any loss to original information. According to the coding principle of lossless compression technique, it can be categorized into two fields: dictionary-based and prediction-based. In dictionary-based, frequently occurring and repetitive patterns are assigned to a shorter codeword. The less efficient codeword is assigned to the others. Based on this principle, the codeword table should be constructed to provide the fixed mapping relationship. Because of this relationship, the prediction technique is utilized to improve coding efficiency. Many methods, including Huffman coding, run length coding, arithmetic coding, LZ77, and LZW, have been widely developed.

Prediction-based algorithms apply prediction technique to generate the residual, and utilize the entropy coding tool to encode it. Many methods, including fast, efficient, lossless image compression system FELICS, context-based, adaptive, lossless image coding CALIC and JPEG-LS have been extensively developed in this field.

1.1 SOFTWARES USED

- Matlab 7 R2008b
- Xilinx ISE 9.2i

1.2 GOAL OF THE PROJECT

The goal of the project is to provide a hardware platform for a VLSI-oriented FELICS algorithm, which can provide more efficient coding principle without data dependency, and maintain competitive coding efficiency. In order to achieve high throughput and reasonable area efficiency,
parallelism and pipelining techniques are adopted in architecture design.

1.3 ORGANISATION OF THE CHAPTERS

Chapter 2 discusses about existing systems and its drawbacks

Chapter 3 deals the description of the project

Chapter 4 explains about the proposed hardware architecture

Chapter 5 discusses the simulation results

Chapter 6 shows the conclusion and future scope of the project

CHAPTER 2
SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Earlier, for image compression system the following codes were attempted,

- HUFFMAN CODING
- RUN LENGTH CODING
- ARITHIMETIC CODING
- LZ77
- LZW

2.1.1 DRAWBACKS IN EXISTING SYSTEMS

HUFFMAN CODING

- If the ensemble changes the frequencies and probabilities change the optimal coding changes
- In text compression symbol frequencies vary with context

RUN LENGTH CODING

- This 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.

ARTHIMETIC CODING

- One is that the whole codeword must be received to start decoding the symbols, and if there is a corrupt bit in the codeword, the entire message could become corrupt.
- Another is that there is a limit to the precision of the number which can be encoded, thus limiting the number of symbols to encode within a codeword.

LZ77

- Not the optimum compression ratio.
- Actual compression is hard to predict.

CHAPTER 3
PROJECT DESCRIPTION

3.1 OVERVIEW OF THE PROJECT

In this project, the image is converted into the binary values by using Matlab version 7.9, then these binary values are used for lossless compression in Xilinx.

The proposed hardware architecture mainly consists of prediction template module, intensity processing module,line buffer, simplified adjusted binary code, Golomb–Rice code with storage-less k parameter selection, and bit stream generator.

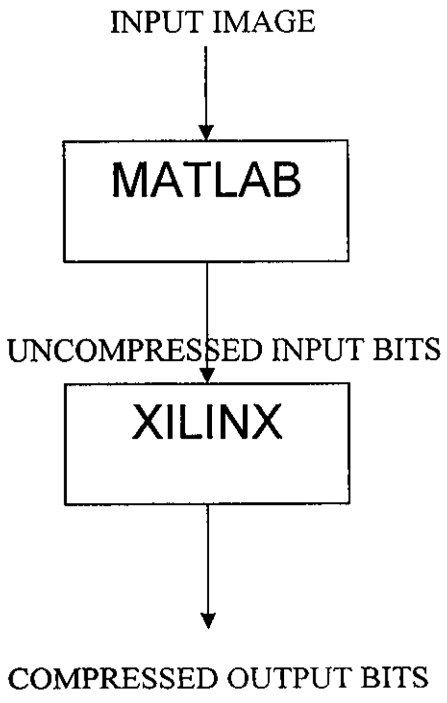


Fig 3.1 Flowchart of overview of the project

CHAPTER 4
PROPOSED HARDWARE ARCHITECTURE

4.1 VLSI ORIENTED HARDWARE ARCHITECTURE

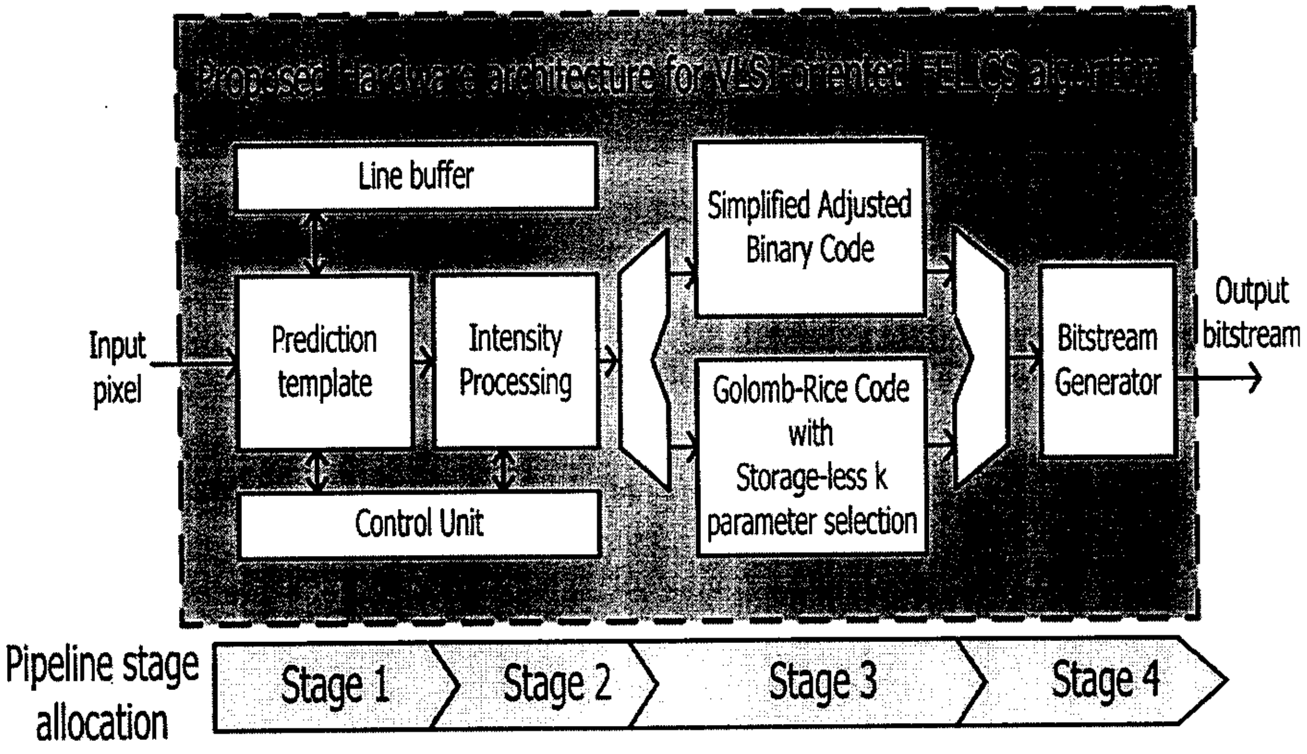

Proposed Hardware architecture for VLSI-oriented FELICS algorithm
Line buffer
Input pixel
Prediction template
Intensity Processing
Control Unit
Simplified Adjusted Binary Code
Golomb-Rice Code with Storage-less k parameter selection
Bitstream Generator
Output bitstream
Pipeline stage allocation
Stage 1
Stage 2
Stage 3
Stage 4

Fig.4.1. Block diagram of the proposed hardware architecture of VLSI-oriented FELICS algorithm

The proposed hardware architecture mainly consists of prediction template module, intensity processing module, simplified adjusted binary code, Golomb–Rice code with storage-less k parameter selection, and bitstream generator.

Data dependency has been completely removed by proposed algorithm, the pipeline stage can be inserted among primary blocks to further improve processing speed. The block diagram of proposed architecture with four-stage pipelining is illustrated in Fig. 4.1.

The prediction template module adaptively adjusts the data path and prepares relative reference pixels for following stage. Subsequently, the intensity processing module identifies which coding mode should be selected and generates the corresponding residual, such as P-L for adjusted binary code and P-H-1 or L-P-1 for Golomb–Rice code. Since the current pixel is encoded with either adjusted binary code or Golomb–Rice code, both corresponding modules are interconnected in parallel with individual data flow.

In addition, the line buffer is applied to store relative reference pixels for each current pixel, and the capacity of it is exactly identical to the width of display resolution. Since both adjusted binary code and Golomb–Rice code belong to variable-length code (VLC), the bitstream generator is responsible to pack them into fixed bit length for transmission on dedicated bus width. The control unit precisely schedules each primary block to accomplish complete encoding flow.

4.2 PREDICTION TEMPLATE MODULE

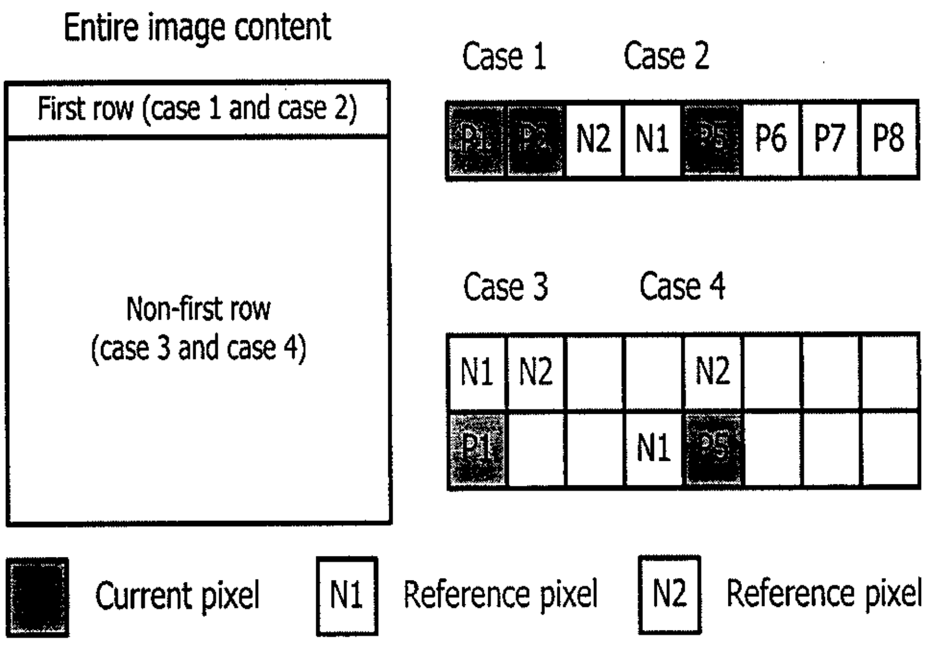


Fig.4.2. Illustration of prediction template in FELICS.

The prediction template module is responsible to make the reference pixel available for each current pixel. The prediction template module adaptively adjusts the data pathfor case 1 to case 4, and prepares relative reference pixels forfollowing stage.

The relationship between reference pixels and the current pixel can be categorized into four cases as follows.

Case 1 Since surrounding reference pixels are not available for the first two pixels, *P1* and *P2*, they are directly packed into bitstream with original pixel intensity.

Case 2 For pixels in the first row except case 1, successive pixels, *N1* and *N2*, are regarded as reference pixels.

Case 3 For the first pixel in each non-first row, no neighbor pixel in horizontal can be found. Thus two upper pixels of *P1*, *N1* and *N2*, are chosen to perform reference information.

Case 4 Otherwise, the left and the upper pixels of current pixel are chosen to perform reference information.

To construct accurate predictors, the information of previous row is required. Thus FELICS utilized a row buffer equal of image width to store the pixels in previous row. In the case 4, the most occurred case, both pixels in horizontal and vertical direction are performed to predict the current pixel. Hence the generated prediction error is significantly reduced to acquire well coding efficiency.

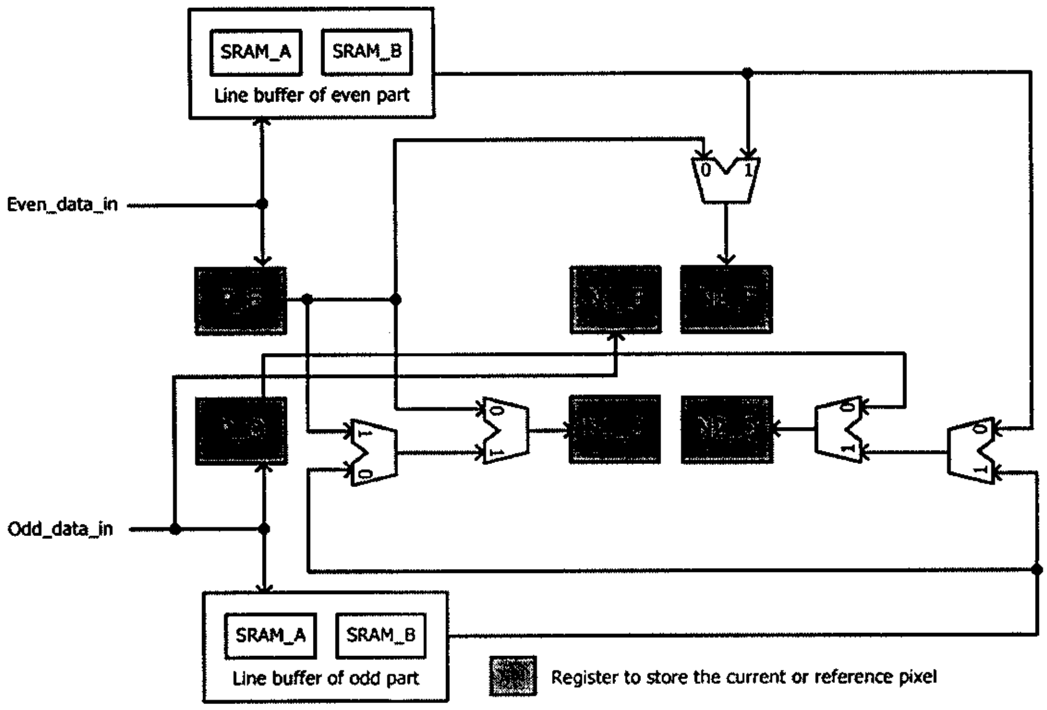


Fig. 4.3.Proposed architecture of prediction template module.

The register stores each current and reference pixel for following intensity processing module. These current and reference pixels are used to following intensity processing module unit.

4.3 INTENSITY PROCESSING MODULE

After the procedure of prediction template, two reference pixels are chosen to predict the current pixel. The smaller reference pixel is represented as *L*, and the other one is *H*.

As depicted in Fig.4.4, the intensity distribution model is exploited to predict the correlation between current pixel and reference pixels. In this model, the intensity that occurs between *L* and *H* is with almost uniform distribution, and regarded as *In-range* with higher probability. The intensity higher than *H* or smaller than *L* is regarded as *Above-range* and *Below-range*, respectively.

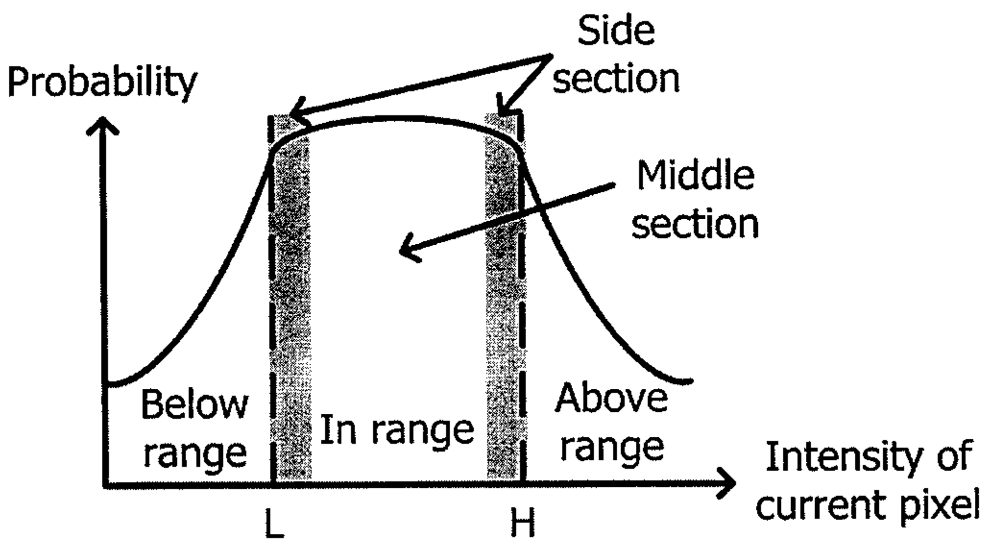
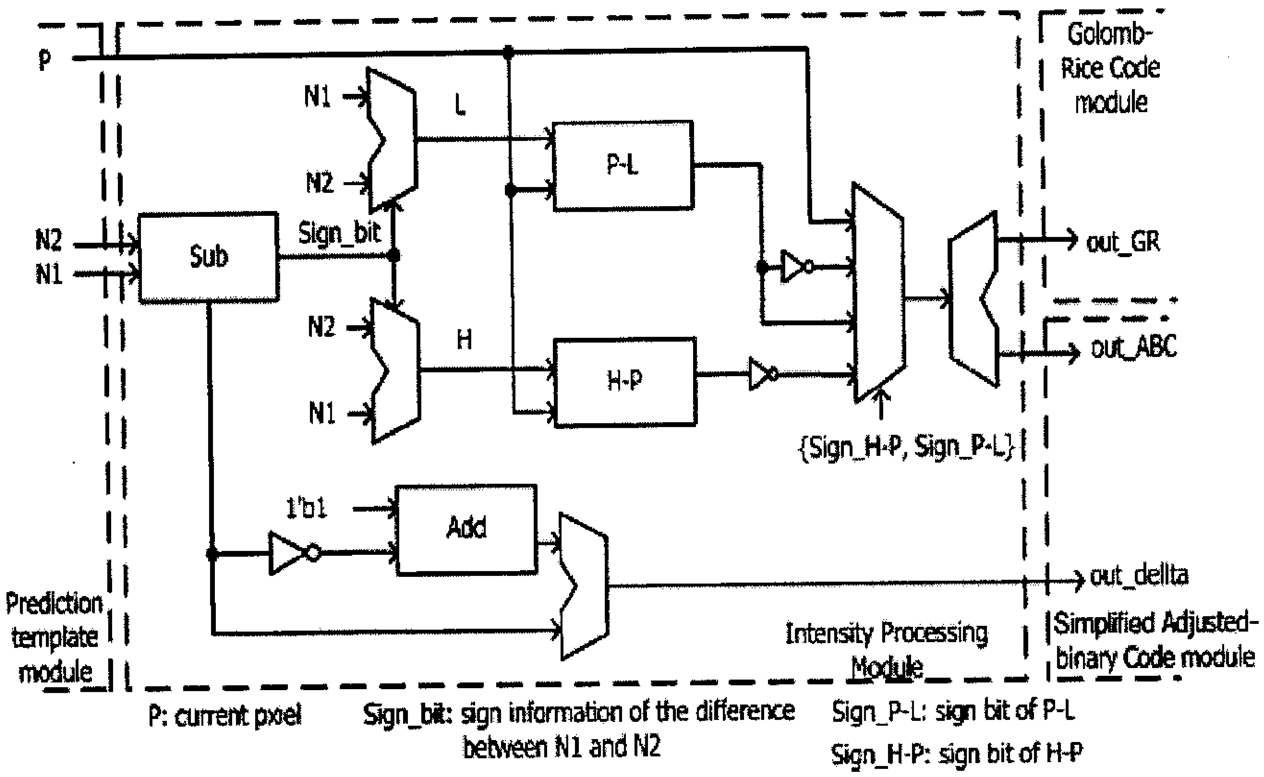


Fig.4.4. Probability distribution model of intensity

And the exponential probability distribution is assigned to both. For *In-range*, the adjusted binary code is selected, and Golomb-Rice code is for both *Above-range* and *Below-range*. The coding mode decision in FELICS is depicted as shown in (2.1). FELICS adopts different predictors according to the intensity distribution of current pixel *P* to provide smaller prediction errors, hence coding efficiency is well-achieved. The relationship of prediction errors and the corresponding intensity distribution is shown in (2.2).

$$\text{Coding mode} = \begin{cases} \text{Adjusted binary code} & \text{if } L \le P \le H \\ \text{Golomb-Rice code} & \text{otherwise} \end{cases} \qquad \text{...........(2.1)}$$

$$\text{Prediction error} = \begin{cases} P-L & \text{if } L \le P \le H & \text{In Range} \\ P-H-1 & \text{if } P > H & \text{Above Range} \\ L-P-1 & \text{if } P < L & \text{Below Range} \end{cases} \qquad \text{...........(2.2)}$$



P
N2
N1
Prediction template module
Sub
Sign_bit
N1
N2
L
N2
N1
H
P-L
H-P
1'b1
Add
{Sign_H-P, Sign_P-L}
Golomb-Rice Code module
out_GR
out_ABC
out_dellta
Intensity Processing Module
Simplified Adjusted-binary Code module
P: current pixel
Sign_bit: sign information of the difference between N1 and N2
Sign_P-L: sign bit of P-L
Sign_H-P: sign bit of H-P

Fig.4.5. Proposed architecture of intensity processing module.

At first, the difference betweenN1 and N2 is derived through a subtraction, and its sign bit is used to identify which reference pixel is H or L. Once both H and L are obtained,the sign bit of H-P and P-L , Sign H-P and Sign P-L,can be exploited to determine which coding mode should beselected. The intensity processingmodule used toidentifies which coding mode should be selected and generatesthe corresponding residual, such as P-L for adjusted binarycode and P-H-1 or L-P-1 for Golomb–Rice code.For Golomb–Rice code, the residual of P-H-1 and L-P-1 can be derived by P-H and L-P values, according to the following equations:

$$P\text{-}H\text{-}1 = \overline{H\text{-}P}+1\text{-}1 = \overline{H\text{-}P}$$
$$L\text{-}P\text{-}1 = \overline{P\text{-}L}+1\text{-}1 = \overline{P\text{-}L}$$

There is also exactly identical to the residual of adjusted binarycode.Since the current pixel is encoded with either adjusted binary code or Golomb–Rice code, both corresponding modules are interconnected in parallel with individual data flow.

4.4 CODING FLOW OF FELICS

Based on intensity distribution model, adjusted binary code and Golomb–Rice code, the FELICS coding flow can be summarized as following steps.

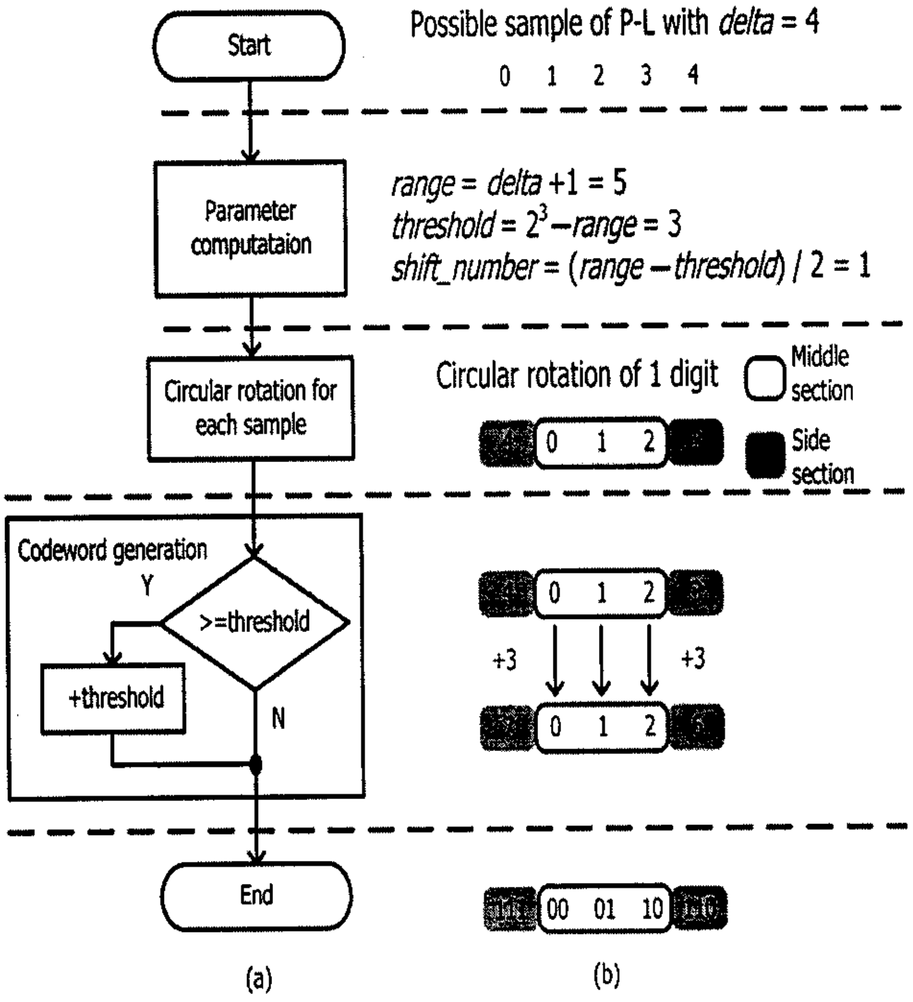

Start
Possible sample of P-L with *delta* = 4
0 1 2 3 4
Parameter computataion
range = *delta* +1 = 5
threshold = 2³–*range* = 3
shift_number = (*range* – *threshold*) / 2 = 1
Circular rotation for each sample
Circular rotation of 1 digit
Middle section
Side section
0 1 2
Codeword generation
>=threshold
Y
N
+threshold
+3
+3
End
00 01 10
(a)
(b)

Fig.4.6. Coding flow of adjusted binary code. (a) Flowchart. (b) Example with delta=4

The first two pixels at first row are directly packed into bitstream without any encoding procedure. According to the prediction template in Fig. 4.1, find the corresponding two reference pixels, N1 and N2.

Assign L=min (N1, N2), H=max(N1,N2) and delta=H-L.

Apply adjusted binary code for P-L in inrange, Golomb–Rice code for L-P -1 in below range, and Golomb–Rice code for P-H -1 in above range.

4.5 LINE BUFFER

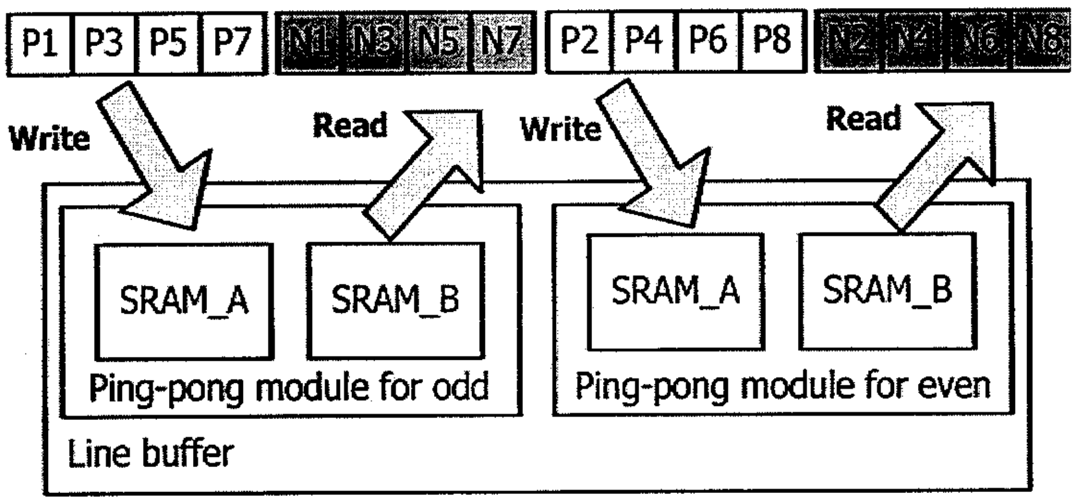


Fig.4.7. Line buffer with ping-pong mode operation.

The line buffer is applied to store relative reference pixels for eachcurrent pixel, and the capacity of it is exactly identical to the width of display resolution. The line buffer should be capable of simultaneously performing write and read operations. Although the dual-port memory can be applied to handle this requirement.In order to support simultaneous accessibility and maintain area efficiency, the static RAM(SRAM) with ping-pong mode operation is adopted in the line buffer.

It indicates that the line buffer should be capable of simultaneously performing write and readoperations. Based on ping-pong operation,each part is further partitioned into two blocks:

SRAM A and SRAM B for individual accessibility. For even and odd engines, the line buffer with ping-pong operation is capable of executing write and read operations at the same time.

4.6 PROPOSED ARCHITECTURE OF SIMPLIFIED ADJUSTED BINARY CODE

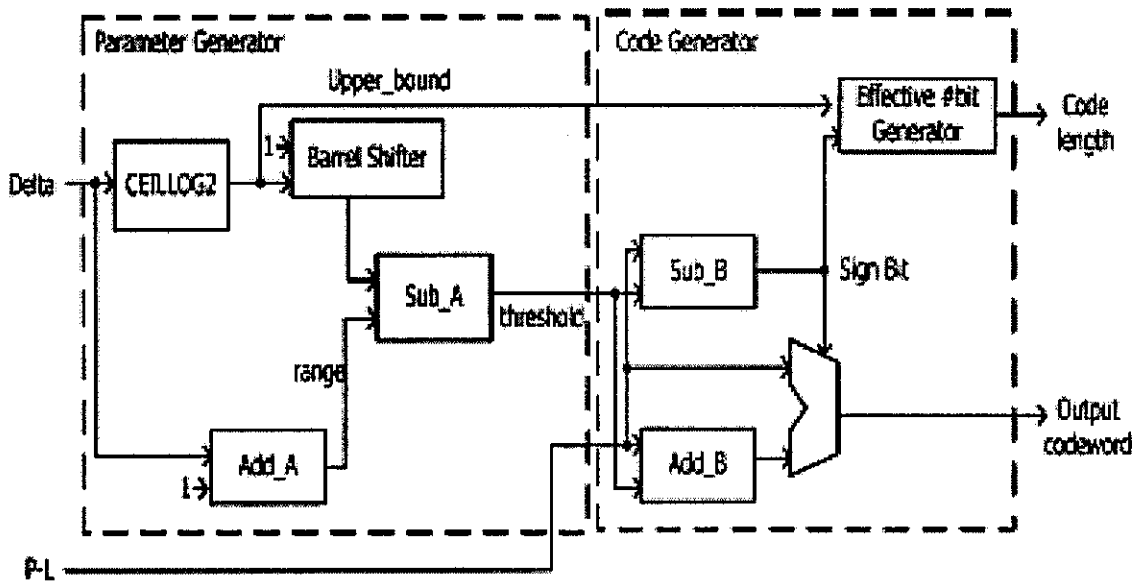


Fig.4.8. Proposed architecture of simplified adjusted binary code module.

The proposed architecture of simplified adjusted binary code module is classified into two parts:

- parameter generator
- code generator

In parameter generator, the Delta, derived from intensity processing module, is passed to CEILLOG2, and it generates upper boundbit information. Concurrently, the range is also obtained by Add_ A With range, the threshold is derived according to following equations :

- delta=H-L,
- range=delta+1,
- upper_bound=log2(range),
- lower_bound=log2(range),
- threshold =2^ upper bound_ range,
- shift_number=(range- threshold)/2

In code generator, one of the possible code words including P-L and P-L + threshold is selected as output symbol. The effective #bit generator indicates the code length for current output code word, and this information is transmitted to bitstream generator module.

4.7 PROPOSED ARCHITECTURE OF GOLOMB–RICE CODE

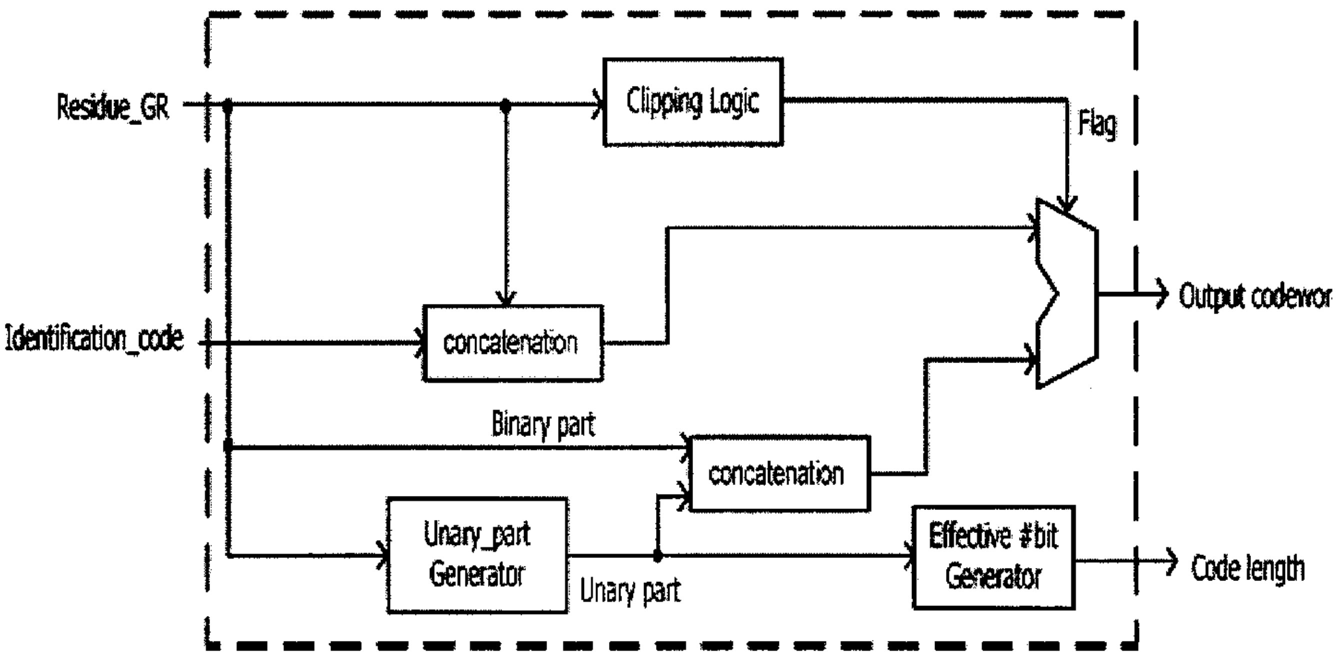


Fig.4.9. Proposed architecture of Golomb–Rice code module with storage-less *k* parameter selection.

For both above range and below range, the probability distribution sharply varies with exponential decay rate, Therefore, Golomb–Rice code is adopted as the coding tool for both above range and below range.

The codeword of sample x is partitioned into unary and binary parts :

Golomb code = Unary Part : [x/m]
Binary Part : x mod m

Where, m is a positive integer, and dominates the coding efficiency of Golomb code. If m is assigned to the power of 2, this coding scheme is regarded as Golomb–Rice code. Its unary and binary part are listed as follows:

Golomb rice code = Unary Part : [x/2^k]
Binary Part : x mod 2^k

Where, k is a positive integer.

The unary part generator is responsible to yield the code word of unary part for a given residual. Based on storage-less parameter selection, the k-parameter is directly set to 2. The binary part can be directly derived by the least K bits of residual and concatenated with unary part to achieve complete code word. If the residual is too large, Golomb–Rice code still presents less efficiency code word. Therefore, the residual is compared with clipping value by clipping logic. If the residual is smaller than clipping value, the Golomb–Rice code is applied. Otherwise, the residual is directly concatenated with Identification code to generate the output code word.

Once the number of bit in unary part is obtained, the total number of bit can be calculated by effective #bit generator. Since the binary part fixedly consumes 2 bits for k=2 . The total code length of Golomb-Rice Code is the summation of number of bit in unary part, binary part and 1 bit for the distinction between both parts.

4.8 BITSTREAM GENERATOR

The bitstream generator is an adaptor between the proposed architecture and external transmission bus or other connection interfaces. The main purpose of bitstream generator is to pack each output code word into bitstream with prefix code and provide an alignment of output bitstream to fit a dedicated bus width.

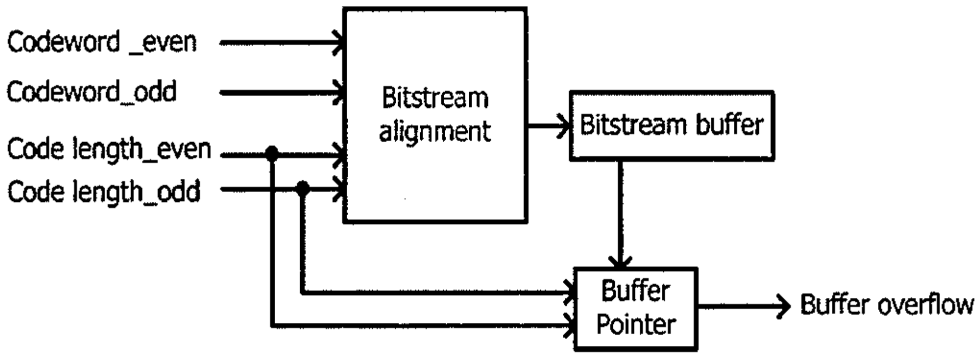


Fig.4.10. Proposed architecture of bitstream generator.

For each code word, the prefix code is inserted into the beginning of it and utilized to identify which coding mode is adopted for this code word. The prefix code is "0" for adjusted binary code, "10" and "11" for Golomb–Rice code in below range and above range, respectively.

The bitstream generator is responsible to pack variable-length code (VLC) into fixed bit length for transmission on dedicated bus width. The bitstream receives the codeword from preceding stages, and packs them into bitstream. In addition, the codeword length provides the information for codeword concatenation under dedicated bus width. Concurrently, the buffer pointer accumulates the codeword length derived from preceding stage to check the status of bitstream buffer. Once the bitstream data are removed from bitstream buffer, the buffer pointer is automatically updated. If the bitstream buffer is full, the bitstream pointer sends overflow signal to inform external systems for the overflow protection procedure.

CHAPTER 5
RESULTS & DISCUSSION

The simulation of this project has been done using MODELSIM 6.2b and XILINX ISE 9.2i.

Modelsim provides a comprehensive simulation and debug environment for complex ASIC and FPGA designs. Support is provided for multiple languages including Verilog, SystemVerilog, VHDL and SystemC.

The simulator software verifies the functionality and timing of the design. The simulator interprets VHDL or Verilog code into circuit functionality and displays logical results of the described HDL to determine correct circuit operation. Simulation allows the designer to create and verify complex functions in a relatively small amount of time.

The ISE software includes Xilinx Synthesis Technology (XST), which synthesizes VHDL, Verilog, or mixed language designs to create Xilinx-specific netlist files known asNGC files. Unlike output from other vendors, which consists of an EDIF file with an associated NCF file, NGC files contain both logical design data and constraints. XST places the NGC file in the project directory and the file is accepted as input to the Translate (NGD Build) step of the implement design process.

5.1 SIMULATION RESULT

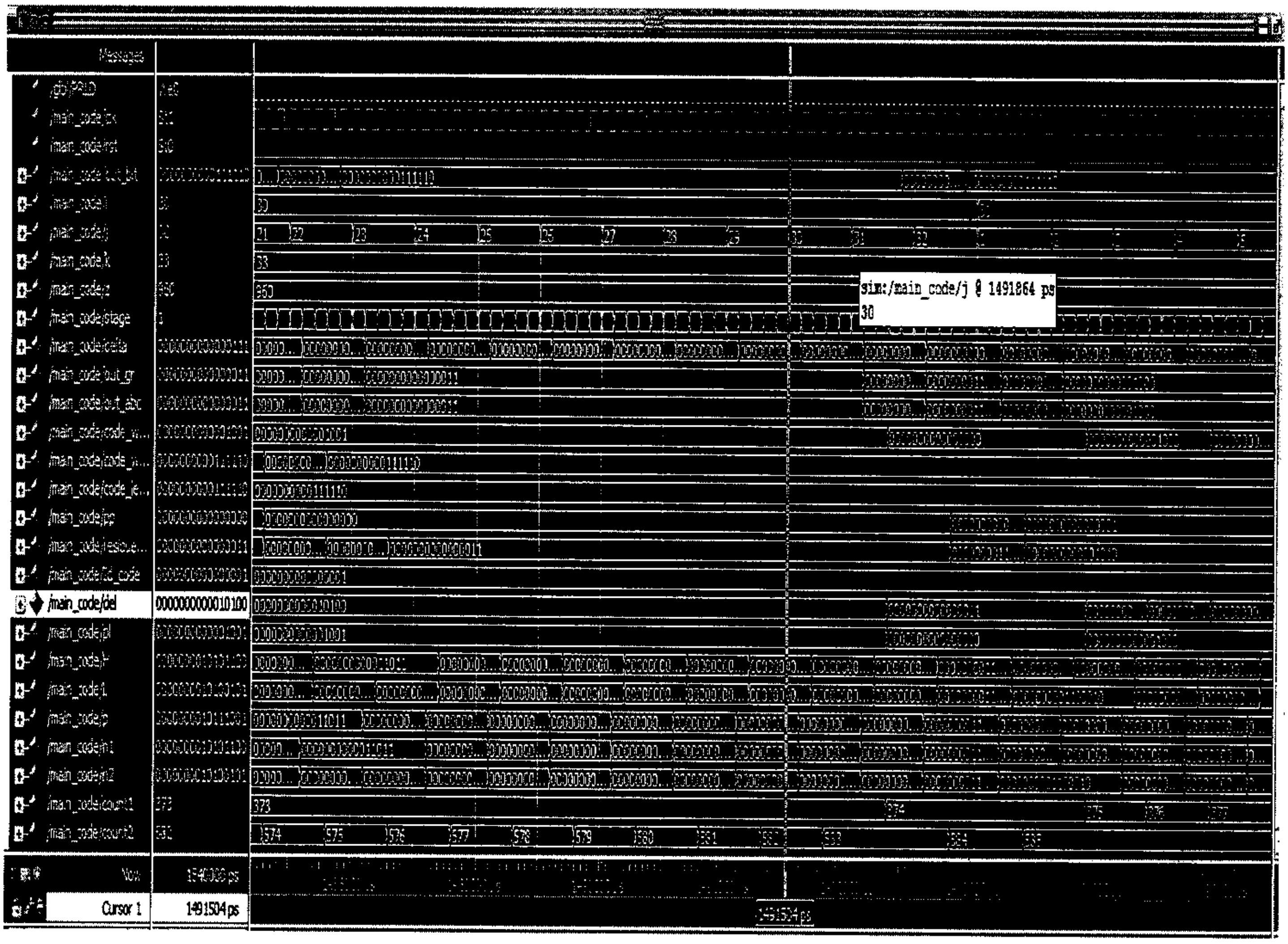
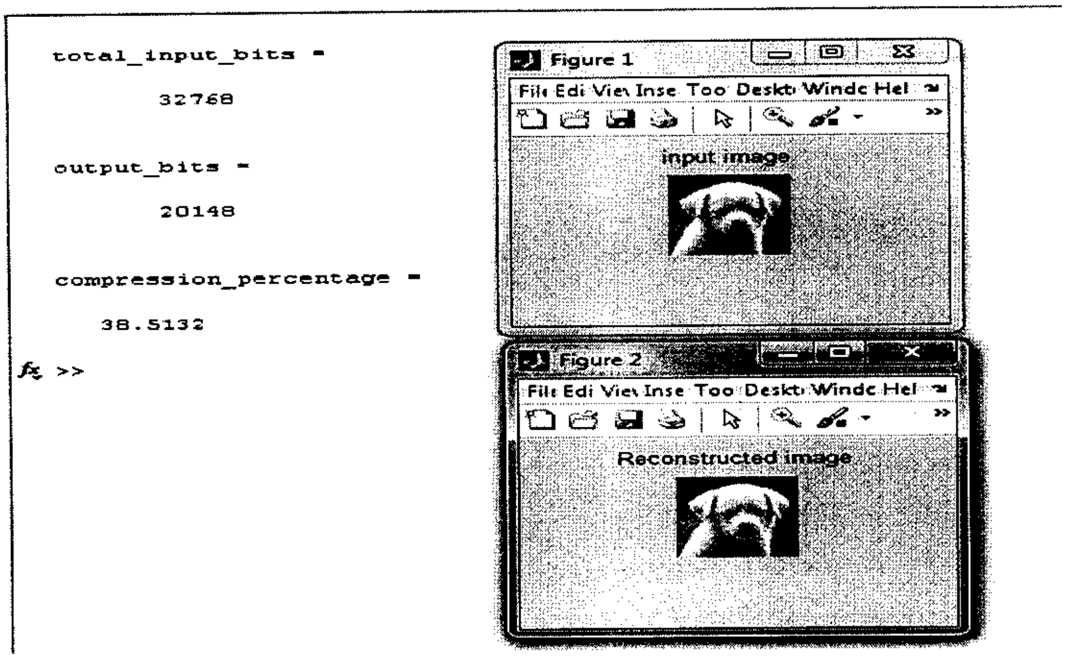
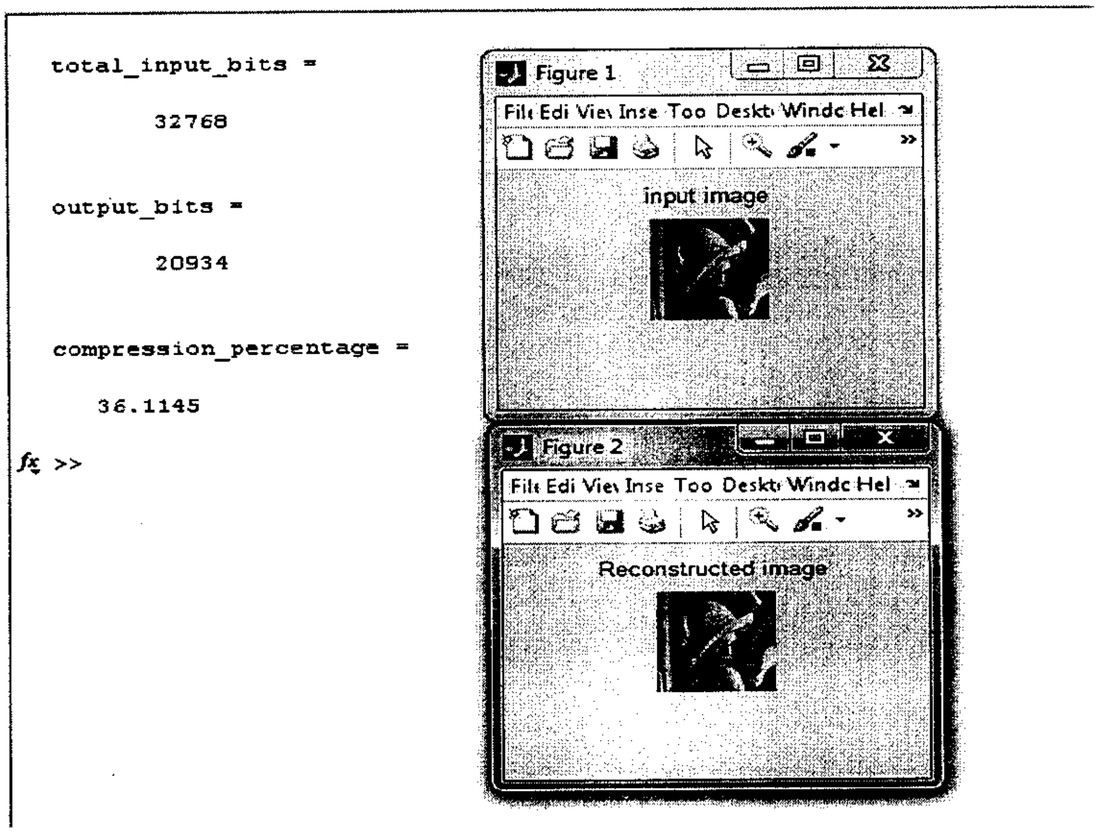


Fig.5.1 Simulation result of VLSI oriented FELICS algorithm

5.2 CODING EFFICIENCY


```
total_input_bits =

        32768

output_bits =

        20148

compression_percentage =

    38.5132

fx >>
```
Figure 1
Fil Edi Vie Inse Too Deskt Windc Hel
input image
Figure 2
Fil Edi Vie Inse Too Deskt Windc Hel
Reconstructed image


```
total_input_bits =

        32768

output_bits =

        20934

compression_percentage =

    36.1145

fx >>
```
Figure 1
Fil Edi Vie Inse Too Deskt Windc Hel
input image
Figure 2
Fil Edi Vie Inse Too Deskt Windc Hel
Reconstructed image

5.3 RESULT SUMMARY

IMAGES USED	TOTAL NO OF INPUT BITS	TOTAL NO OF OUTPUT BITS	% OF COMPRESSION	NO OF PIXELS USED BY [illegible] CODE	NO OF PIXELS [illegible] CODE
	32768	20934	36.1145%	378	644
	32768	20148	38.5132%	382	640

INTERPRETATION OF RESULTS

The input image is converted into binary values by using Matlab, then these binary values are given as input to the lossless compression engine, which utilizes VLSI oriented FELICS algorithm. By this algorithm the given input bits have been compressed. The algorithm mainly consists of simplified adjusted binary code and Golomb–Rice code.

Simulation results shows that number of pixels used by both the codes and the pixels of compressed output.

For getting the compression percentage the same codes have been used in Matlab and achieved around 38% of compression.

CHAPTER 6
CONCLUSION

6.1 CONCLUSION

In this project, the VLSI-oriented FELCIS algorithm is implemented with two main techniques, simplified adjusted binary code and Golomb–Rice code. The number of sequential arithmetic operation is significantly reduced in simplified adjusted binary code and hence processing speed is also improved. For Golomb–Rice code, the storage-less k parameter selection removes the data dependency, and the storage for entire cumulation-table is eliminated.

In comparison with original FELICS algorithm, the Compression Ratio is improved about 35%. Based on VLSI-oriented FELICS algorithm, two-level parallelism and four-stage pipelining were adopted in proposed hardware.

6.2 FUTURE ENHANCEMENT

The future scope of this project is to implement the proposed architecture with superior parallelism efficiency. Moreover, with multilevel parallelism, the proposed architecture can be further applied in more advanced HD display specifications such as QHD and QFHD.

In High Definition (HD) display applications, encoding capability can be achieved with a high-quality specification with complete red, green, blue color components.

REFERENCES

[1] P.-C. Tseng, Y.-C. Chang, Y.-W. Huang, H.-C. Fang, C.-T. Huang, and L.-G. Chen, "Advances in hardware architectures for image and video coding-a survey," *Proc. IEEE*, vol. 93, no. 1, pp. 184–197, Jan. 2005.

[2] S.-Y. Chien, Y.-W. Huang, C.-Y. Chen, H. H. Chen, and L.-G. Chen, "Hardware architecture design of video compression for multimedia communication systems," *IEEE Commun. Mag.*,

[3] T.-M. Liu, T.-A. Lin, S.-Z. Wang, and C.-Y. Lee, "A low-power dualmode video decoder for mobile applications," *IEEE Commun. Mag.*, vol. 44, no. 8, pp. 119–126, Aug. 2006.

[4] T.-C. Chen, S.-Y. Chien, Y.-W. Huang, C.-H. Tsai, C.-Y. Chen, T.-W. Chen, and L.-G. Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder," *IEEE Trans. Circuit Syst. Video Technol.*, vol. 61, no. 6, pp. 673–688, Jun. 2006.

[5] D. Huffman, "A method for the construction of minimum redundancy codes," *Proc. IRE*, vol. 140, pp. 1098–1011, Sep. 1952. [10] S. Colomb, "Run length encoding," *IEEE Trans. Inf. Theory*,

[6] G. G. Langdon, "An introduction to arithmetic coding," *IBM J. Res.Dev.*, vol. 28, no. 2, pp. 135–149, Mar. 1984.

[7] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory*, vol. IT-23, no. 2, pp. 399–401, May 1977.

[8] T.Welsh, "A technique for high-performance data compression," *IEEE Comput.*, vol. 17, no. 6, pp. 8–10, Jun. 1984.