Wireless Pers Commun (2016) 90:1291–1315
DOI 10.1007/s11277-016-3391-9

Hardware Implementation of Architecture Techniques for Fast Efficient Lossless Image Compression System

N. Muthukumaran[1] · **R. Ravi**[1]

Published online: 2 June 2016

Abstract In this research paper, a high throughput memory efficient pipelining architecture for Fast Efficient Set Partitioning in Hierarchical Trees (SPIHT) image compression system is explained. The main aim of this paper is to compress and implement the image without any loss of information. So, we are using spatial oriented tree approach in Fast Efficient SPIHT algorithm for compression and Spartan 3 EDK kit for hardware implementation analysis purpose. Integer wavelet transform is used for encoding and decoding process in SPIHT algorithm. Here, we are using pipelining architecture to implement it in FPGA kit because pipeline architecture is more suitable for hardware utility purpose. Generally an image file will occupy more amount of space. In order to reduce the memory size no loss during transmission we are using this approach. By this way we are attained maximum PSNR value, CR and also produced very high accurate image after decompression, when compared with the results of other previous algorithms. In this module, the hardware tools used are dual core processor and FPGA Spartan 3 EDK kit and the software tool windows 8 operating system and the tool kit is MATLAB 7.8.

Keywords Image compression · Compression ratio · Spatial oriented trees · Fast efficient SPIHT · VLSI pipeline architecture · FPGA kit Implementation

1 Introduction

The fundamental standard of image or data compression is decreased dimension and its representation information in the form of compressed image. In image compression storage, space is important because less memory space means less time required to process the

✉ N. Muthukumaran
kumaranece@gmail.com

R. Ravi
fxhodcse@gmail.com

[1] Francis Xavier Engineering College, Tirunelveli, Tamilnadu 627003, India

image [1, 2]. In image compression, it becomes still more important because to transfer the uncompressed image it requires more space and bandwidth. In order to occupy less memory space, we are going for image compression technique [4, 6]. Here, we are using fast efficient SPIHT algorithm for image compression process. The generally used compressed files are 'win rar' and 'win zip' files which are used to compress the text files [9]. In order to compress the image files and regain without any loss in the image we are going for SPIHT algorithm [13]. Using this algorithm, we can compress and decompress an image without any loss in the image pixels. Generally, two types of image compressions are used. One is lossy compression and another one is lossless compression [3].

The architecture used is pipeline architecture. It is a hardware utility architecture and easy to design which is shown in Fig. 3. In pipeline architecture several stages are used to implement the design technique [16]. The main contribution of this architecture is, a simple context state model which is based on the neighbour pixel's significant states is designed for hardware implementation. In order to achieve high speed architecture, we adopt a fixed breadth first search scan order for SPIHT coding instead of variable scan order to avoid rescanning the wavelet coefficients [10, 11].

According to the context model, different context symbols formed by SPIHT algorithm are processed in the pipelining process for speedup purpose [5]. In order to improve the throughput of our pipeline, we utilize two methods to remove the bottlenecks in the whole image coding process. One method is a bit-plane parallel scheme for all wavelet coefficient bit planes, which changes the process order of bit-plane from sequential to parallel manner. The other important way is an out of order mechanism for the execution in the pipeline [14, 16]. In order to reduce memory size, an internal memory array is used for the cumulative probability variables. Since the architecture uses an FPGA platform as prototype, plenty of flip-flops in the device can be used for the memory array instead of external RAM structure.

2 Problem Formulations

The fundamental scheme of the wavelet transform is to be characterized by the mother wavelet function. Here DCT is applied to compress the image and also SVD, JPEG, MPEG compression are used [1, 7, 8]. In wavelet based image compression, the coders provide an enhanced progress and the existing methods related to this domain are SPIHT [4], EZW [13], Wavelet transform, Fourier Transform, DFT, JPEG [7], DCT based coding which are the ancient methods and have many disadvantages such as compression range is not up to the level, PSNR range is low, high MSE, lossy compression, complex hardware implementation, high cost and consuming very high power. In order to overcome this, proposed method of problem discussion and analysis section is taken into account.

The Fourier Transform utility lies in its ability to analyze a signal in the time domain for its frequency content. The transform works by first translating a function in the time domain into a function in the frequency content gives the Fourier Coefficients of the Transformed function [12]. The Discrete Fourier Transform (DFT) estimates the Fourier Transform of a function with a finite number of its sampled points. The sampled points are supposed to be typical of what signal looks like at all other times [10, 15]. The DFT has symmetry properties almost exactly the same as the continuous Fourier Transform.

3 Problem Discussion and Analysis

3.1 Flow Steps of DWT Algorithm

Firstly, the wavelet transform is applied to the rows of the image and then, the columns of the image. Thus the image is decomposed into four quadrants with these flow steps as discussed below, For the LL section: The upper left quadrant consists of all coefficients, which were filtered by the analysis low pass filter along the corresponding columns. For the HL/LH section: The lower left and the upper right blocks were filtered along the rows and columns blocks. The LH block contains vertical boundaries and HL block shows horizontal boundaries very clearly [1, 5]. For the HH section: The lower right quadrant was derived analogously to the upper left quadrant by means of the use of the analysis high pass filter. HL, LH, HH have better CR with appreciable loss of information.

3.2 SPIHT Algorithm

The wavelet transform is functional to an image, the main algorithm works by partitioning the wavelet decomposed image into important and unimportant partitions. The SPHIT algorithm based image compression is the image spilt in four parts via row and columns. There are four ways to split the input image such as probable method is, LL (Low to low pass), LH (Low to high pass), HL (High to low pass) and HH (High to high pass) in Fig. 1. There are several properties for the SPHIT algorithm based on the scheme multi resolution scalability with normally a high PSNR value. The progressive step of the properties is the optimized technique for image transmission with fast coding and decoding process. Finally, it can be used for lossless image scheme for compression and efficiently of error protection analysis.

3.3 Proposed Pipeline Architecture System

In Fig. 2 shows the pipeline structural design is a communication line information for everything. Here, a filter is a process for continuously read and write messages from an input pipeline and then writes the result to an output pipeline. Between the input and

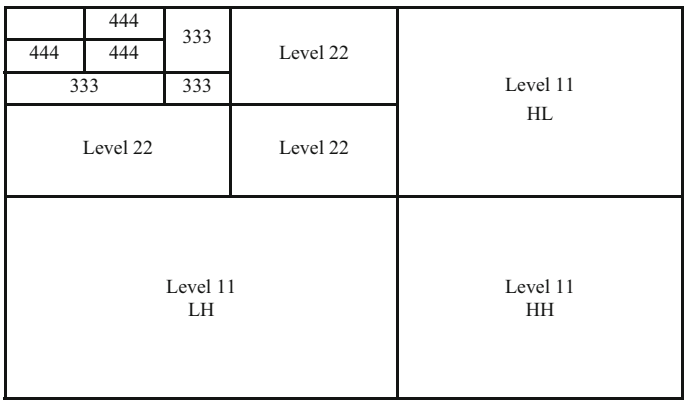

Fig. 1 Image decomposition using spatial oriented trees

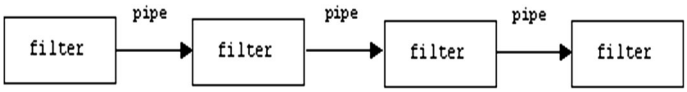

Fig. 2 Performance of pipeline

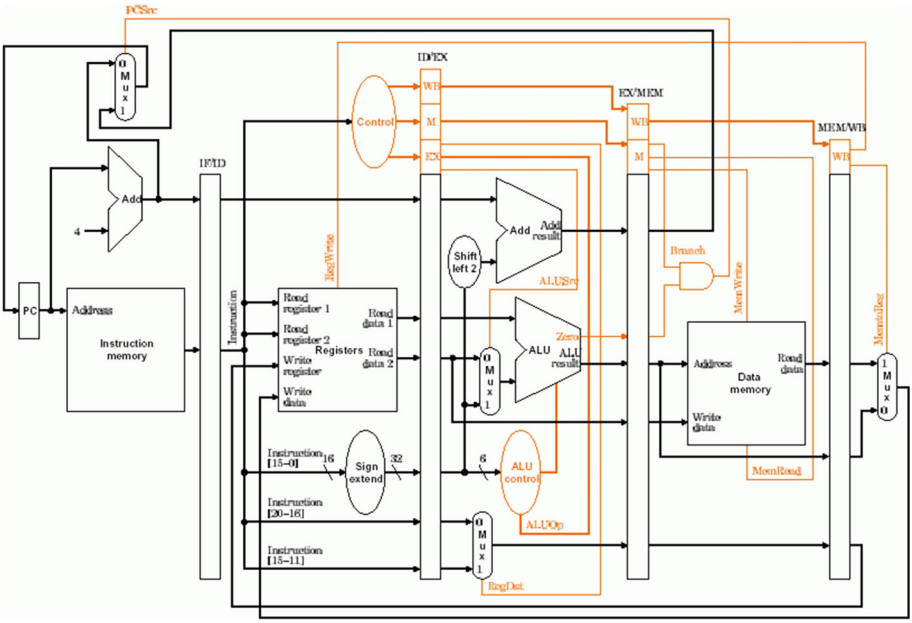

Fig. 3 Pipeline architecture

output, it processes each message at every time of process. Pipeline processes is a sequence of functional units which performs a assignment in several steps of processes. Each functional unit takes input and output values and these values are stored in output buffer field. Here, every data's entered in the sequence flow of first input data is output buffer values of previous stage. These pipelines process performance may be synchronous or asynchronous process of structural design in Fig. 3.

3.4 Compression Parameter Measurement

Compression Ratio = (Original Image Size)/(Compressed Image Size).
Original Image Size = Number of rows * Number of Columns.
Compressed Image Size = Number of bits after encoding.

4 FPGA Kit Implementation and Result Discussion

The paper is implemented in a reconfigurable hardware structure like FPGA and a soft core microcontroller such as micro blaze.

4.1 FPGA Kit Implementation

The implementations in XILINX Platform Studio (XPS) is used to develop Embedded Development Kit based system designs. Designers use XPS to configure and construct a hardware requirement of their embedded systems. The XPS converts the designer's display place requirement into a Register Transfer Level (RTL) justification. Verilog or VHSIC Hardware Description Language (VHDL) writes a set of script to automate the implementation from the beginning RTL to the small piece of stream file. Then the XPS boundary is the equipment which is used right through the whole design flow section. In the hardware description language of VHDL is used in real time circuit design process to digital signal such as FPGA and IC's.

4.1.1 XILINX Platform Studio

The implementations of XPS is an IDE used to develop EDK based system designs. Designers use XPS to organize and put together a hardware requirement of their embedded systems. The XPS converts the designer's platform requirement into a synthesizable RTL explanation. The VHDL coding is used to write set of screenplay to computerize the implementation of fixed system from RTL to the bit stream file. The XPS is a GUI window that helps you to identify your system that is, which processors, memory blocks and other FPGA peripherals to use and how the different peripherals are connected and finally the memory map that is for addresses for memory mapped I/O peripherals.

XPS also interface the tools used throughout the whole design flow. There are three components used in this.

1. Micro blaze processor
2. UART (serial port)
3. Memory block

4.1.2 Process of XPS Tools

The following are steps of XPS process

1. Processor Hardware development
2. Verification File Generation
3. Design Implementation
4. Device Configuration

XPS main windows is divided into these three areas

1. Project Information area- Project IP Catalogue tabs
2. System Assembly view
3. Console Window

It has labels to identify the following areas

1. Connectivity panel
2. View buttons
3. Filters panel

The XPS Tools are,

1. Platform Generation Tool—PLATGEN
2. Simulation Model Generation Tool—SIMGEN

4.1.3 FPGA Internal Structure

The initial FPGAs didn't have internal memories but now a day's all new FPGAs have internal memories with that internal memories a lot of real time applications may be implemented. Generally the FPGA internal structure block illustrations are available to count the number of divide address buses going to the RAM. Every manager has a dedicated address bus and in addition each manager has also a read, a write or both information data buses. If having both information data user always mean and manager can read and write at the same time. Writing and reading to the RAM is usually done synchronously but can also from time to time be done asynchronously. Xilinx has a group of flexibility in the RAM distribution for the reason that it also allows using the logic cells.

4.1.4 Port Techniques

The port techniques are involved in the interfacing of system and FPGA kit. There are two types of ports as discussed below. One is serial port and another one is parallel port as in Figs. 4 and 5.

The serial port is the serial connection of port technique from one terminal to another terminal. The real problem involved in serial port is bandwidth and limited ports connection. The serial port is the slowest of the group compared to the parallel port. Parallel ports connection are easy and faster compared to the serial ports connection. The disadvantage of parallel port is that it needs additional number of transmission lines then only

Fig. 4 USB to serial adapter

Fig. 5 USB to parallel adapter

parallel ports are not used in long distance. Normally serial ports are having two data lines. One is transmission and another one is receive line.

4.1.5 Spartan 3 EDK

Spartan 3 EDK kit is the VLSI parameter analysing and display kit and its low cost Spartan series is artificial manufactured with advanced process development technology. For the display purpose, the Spartan 3E FPGA family is offered and platform description were looking for ideal logic gate designs. The Spartan 3 EDK kit provides a controlling and self progress display place for designing a new Spartan 3 FPGA kit implementation from Xilinx software. In Spartan 3 EDK kit is a user friendly implementation and the key features of on board connection of input and output devices is making the ideal display place to test with any new image compression based design from a straightforward digital logic circuit. There are various key features and kit details and supportive software connections discussed later.

4.1.6 Key Features of Spartan 3 EDK

The key features of Spartan 3 EDK kit are generally having the several types of characteristics involved.

1. 2 no's of seven segment display
2. 2 × 2 no's of matrix keypad interface
3. RS 232 serial port cable
4. 8 no's of digital inputs
5. 8 no's of digital display for LEDs outputs
6. 2 × 16 no's of Character LCD interface
7. Video Graphics Array (VGA) Interface
8. 50 MHz crystal oscillator clock source
9. Onboard connection 5, 3.3, 1.2 V regulators service

4.1.7 Benefits of Supportive Software's

1. It supports VHDL.
2. JTAG programming and debugging the connection.
3. Xilinx ISE and Xilinx EDK.

4.1.8 Details of the Spartan 3 EDK kit

1. Power Adaptor and Parallel JTAG programmer
2. RS 232 Cable connection
3. Devices: XC3S200 (Spartan 3 FPGA)
4. Clock: 50 MHz On-Board crystal

4.2 Results and Discussion

The results and discussion come in two fold over that is software simulation output and hardware implementation analysis. For the simulation output of image compression

techniques is used to measured the compression ratio, MSE and PSNR results for representative images using Fast Efficient SPIHT coding algorithm in Figs. 6 and 7. The result of Fast Efficient SPIHT coding technique is the high accuracy transform with simple framework model. The harware implementation of FPGA based kit module is implemented in FPGA kit with Spartan 3 EDK. We can interface it with the personal computer and compress the images.

In Fig. 8 shows the header file creation for the FPGA kit inter connection purpose and then this header file value is sent to FPGA domain via a serial port device. Figure 9 shows the header file which is thus created for the given input image. It also shows the pixel values of the image. Figure 10 shows the configuration of Microblaze microprocessor with the base system builder which is used for embedded system design. Figure 11 shows the Embedded development kit platform studio view. In this section a new design has been created. Figure 12 shows the representation based on the base system builder with Xilinx Platform Studio devices. Herewith, we have design board name and section. Figure 13 shows the Microblaze 32 bit platform studio snap shot. In this Microblaze have 32 bit devices and using this section we can measure the device, package and the speed grade of the device.

In Fig. 14 shows the microblaze configuration data instruction with the device. Herewith, we have measured the reference clock frequency and processor bus clock frequency. Here the frequency value is 50 MHz and also the interconnect onchip hardware debug module mode is selected. Figure 15 shows the configuration Input and Output (I/O) Interface device. Here, we can select the serial port techniques cable and also select the baudrate value and data rate values are measured. Figure 16 shows the configuration

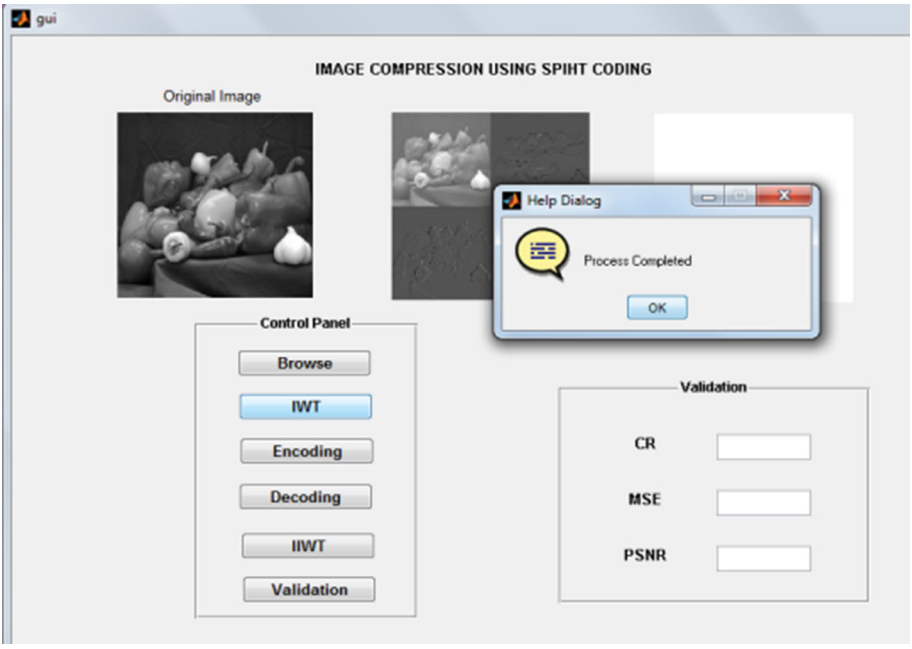

Fig. 6 Compressing the image using wavelet transform

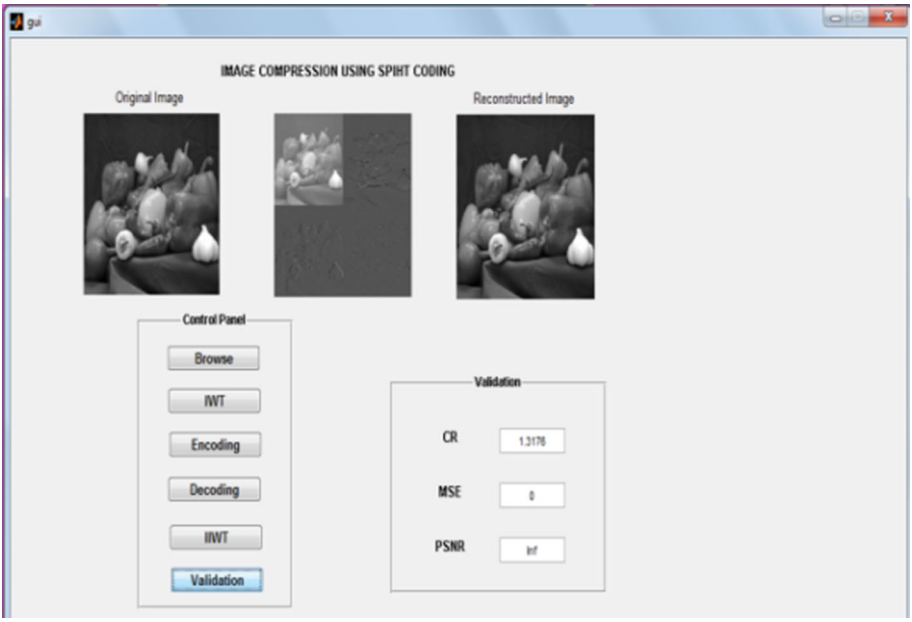

Fig. 7 PSNR and CR values using SPIHT algorithm

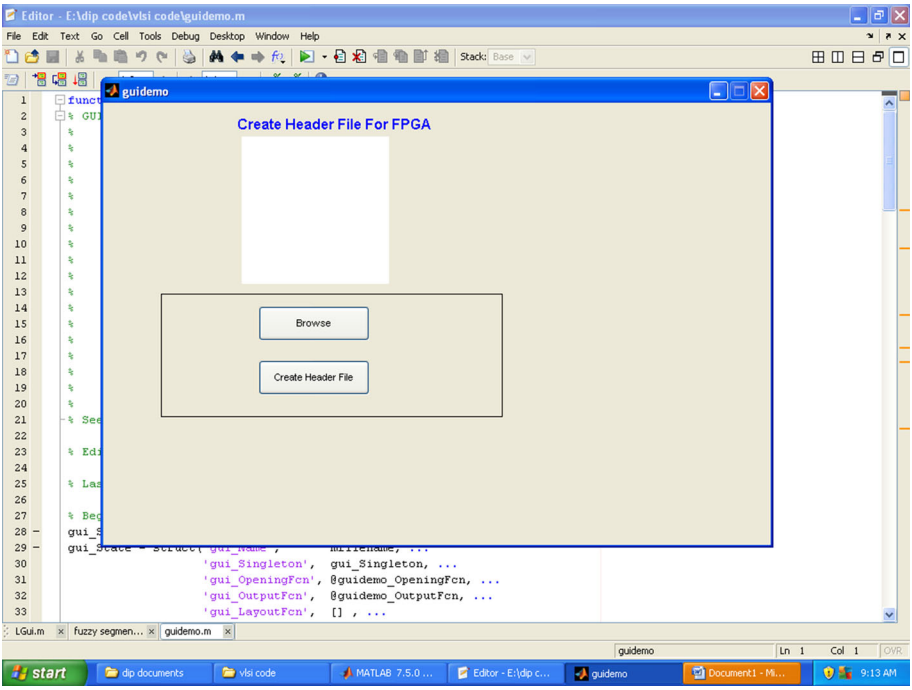

Fig. 8 Header file creation for FPGA

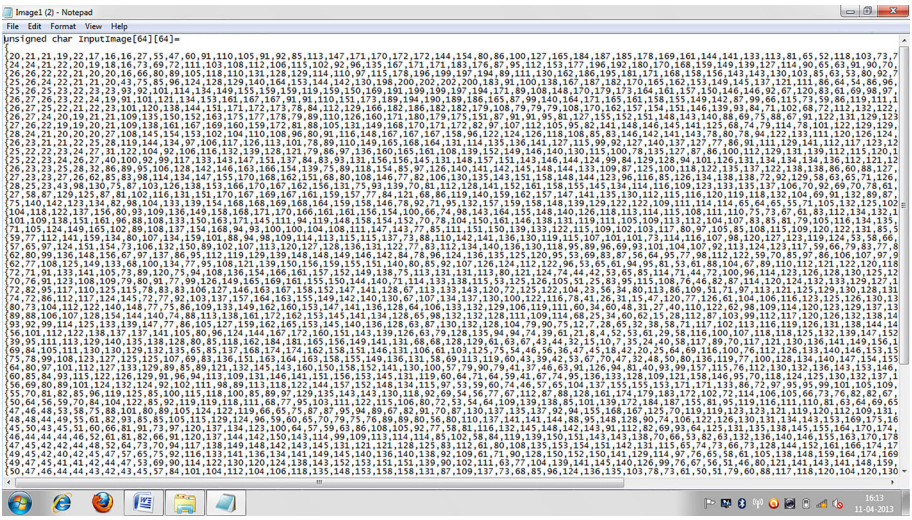

Fig. 9 Header file

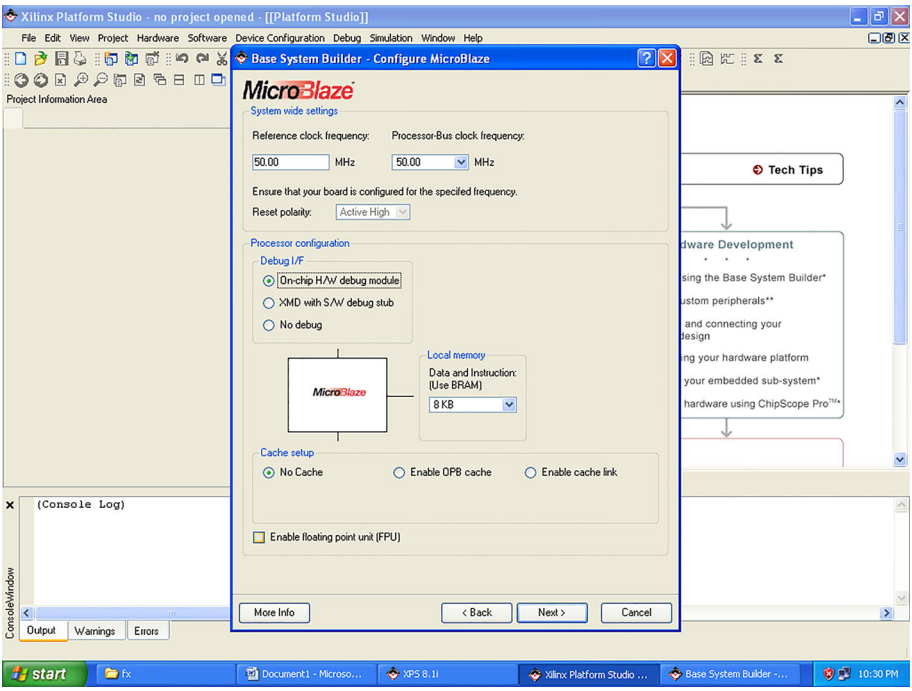

Fig. 10 Microblaze configuration

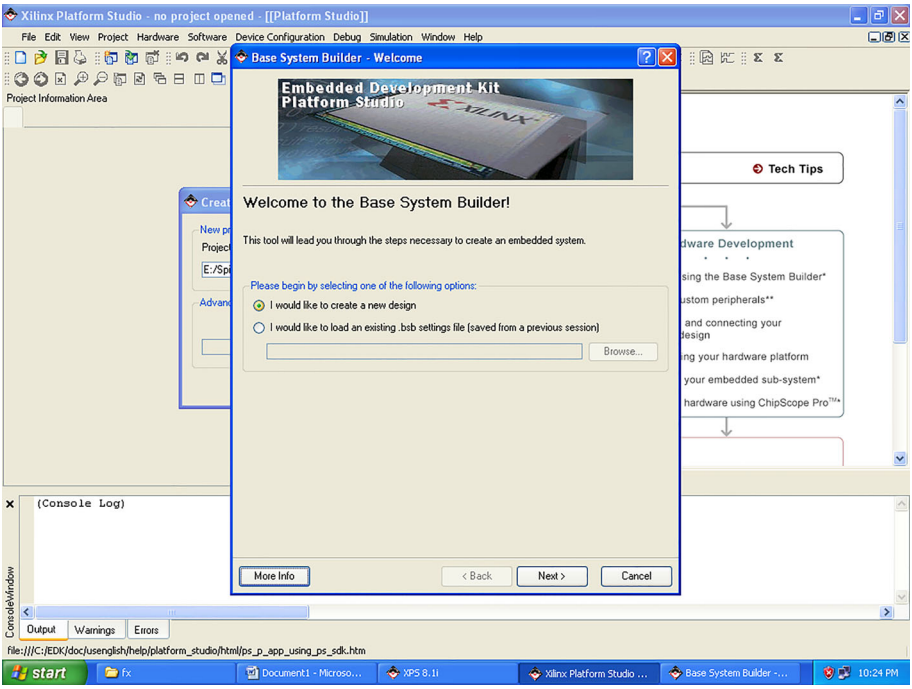

Fig. 11 Embedded development kit platform studio

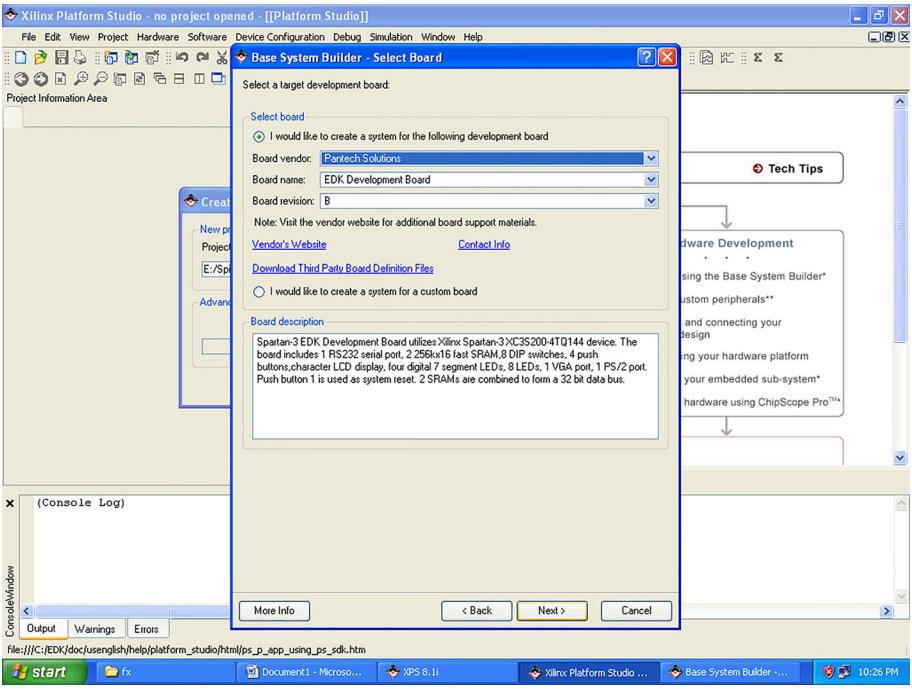

Fig. 12 Xilinx Platform Studio

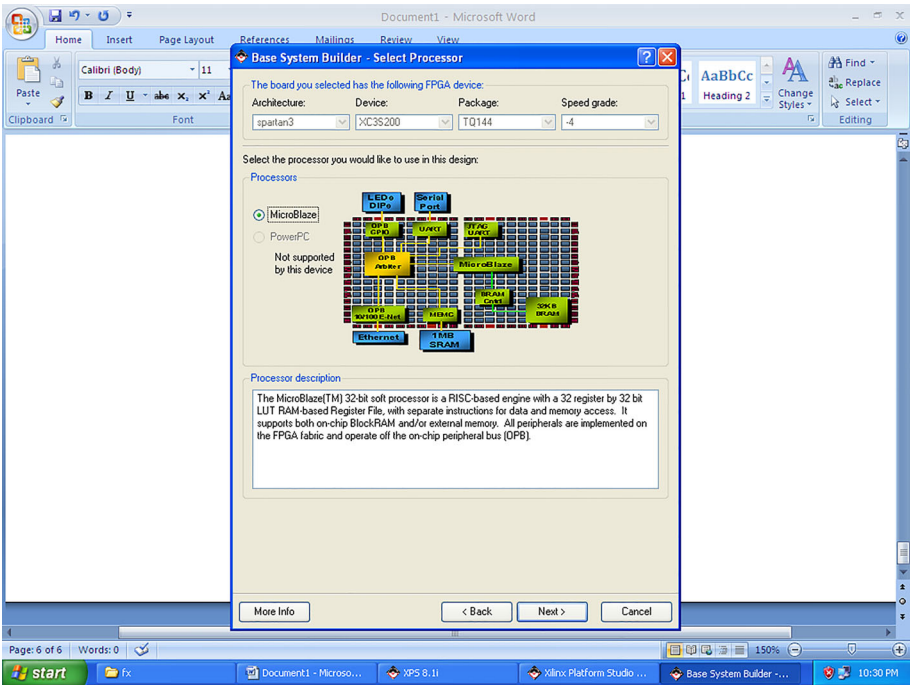

Fig. 13 Microblaze 32 bit Platform Studio

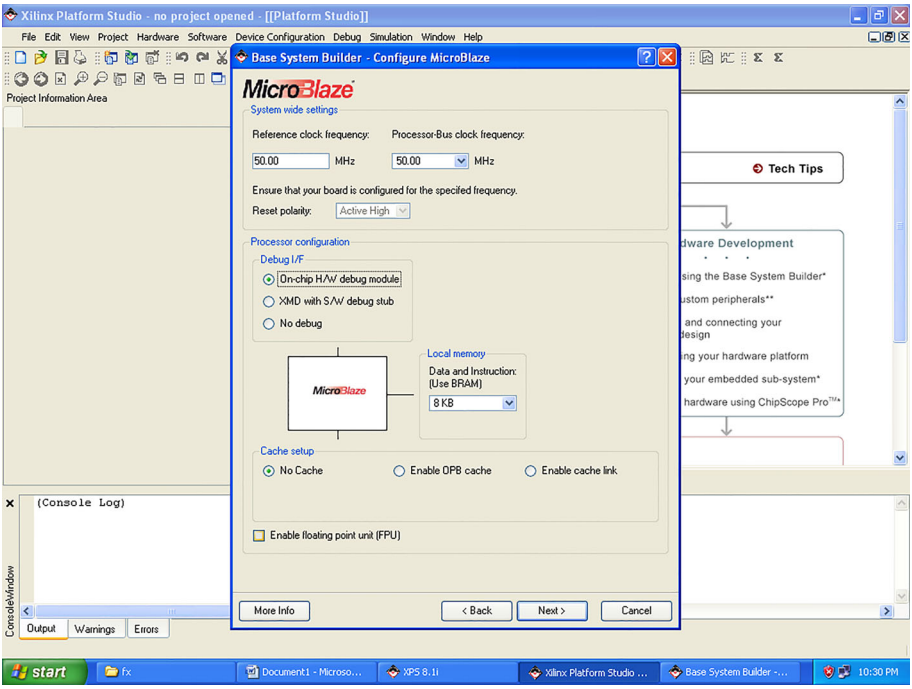

Fig. 14 Microblaze configuration data instruction

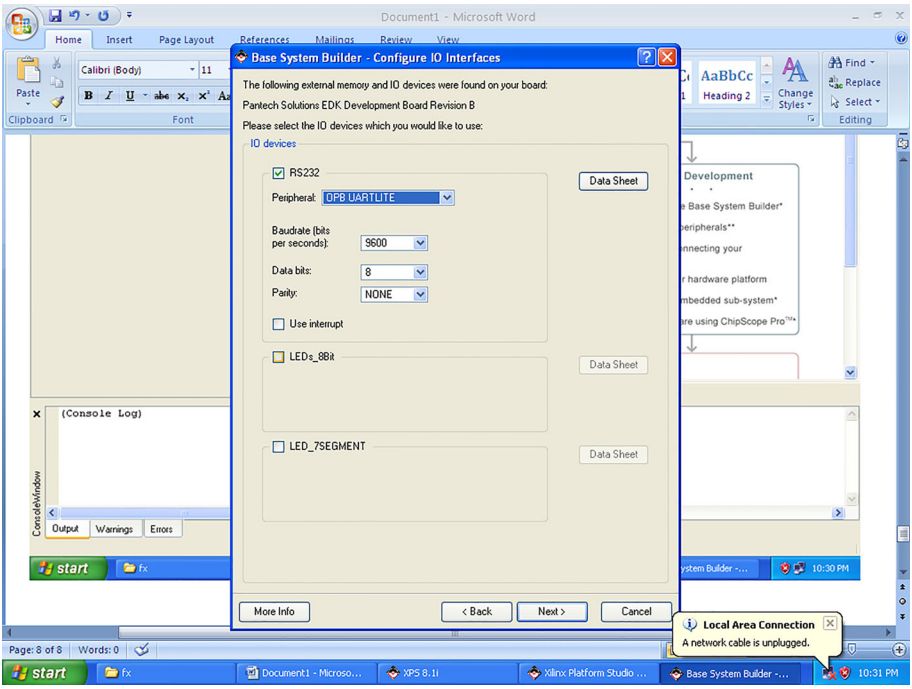

Fig. 15 Configuration I/O interface

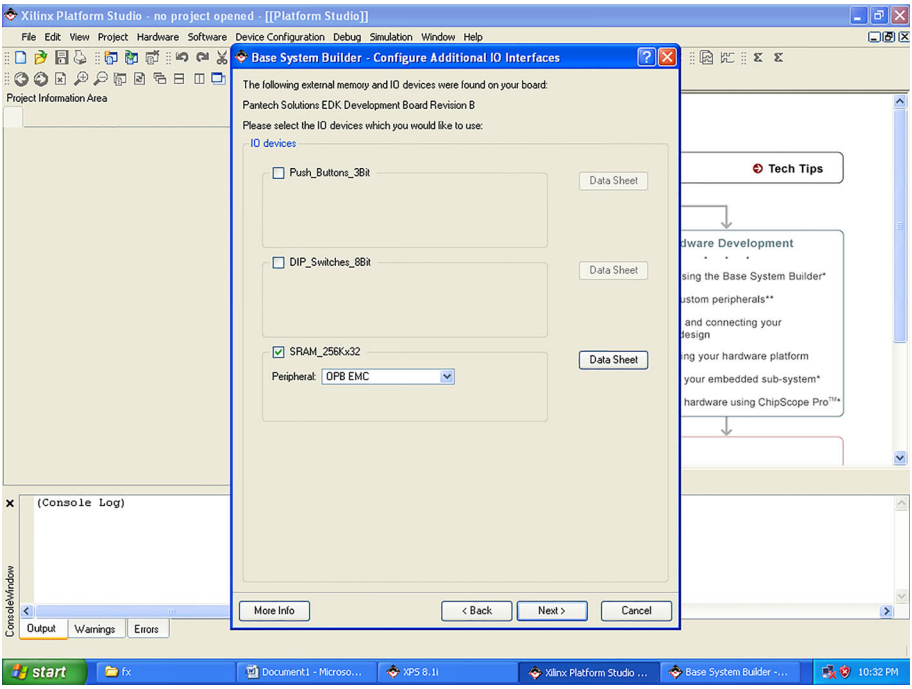

Fig. 16 Configur additional I/O interface

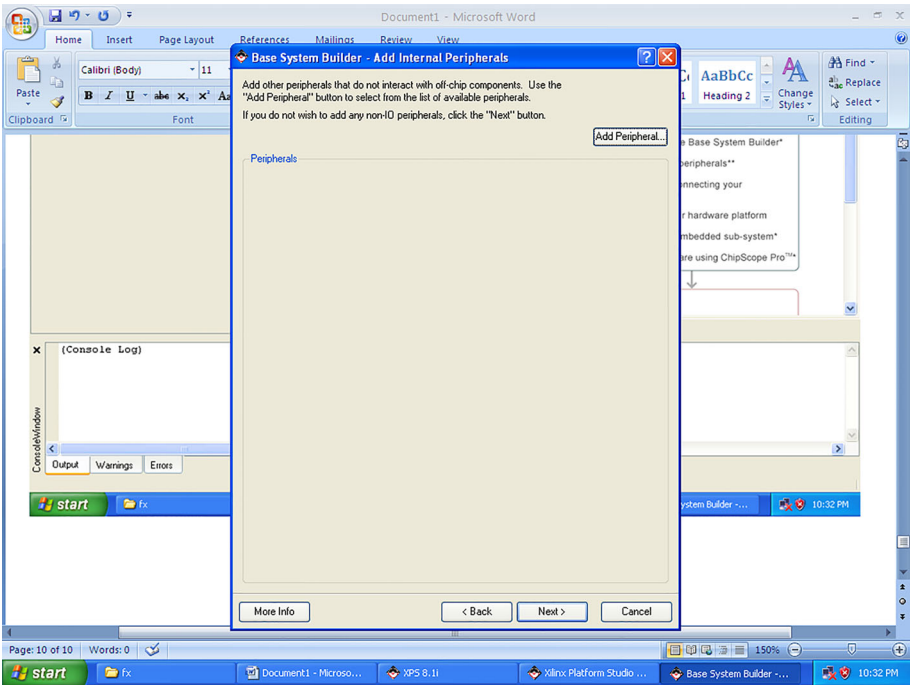

Fig. 17 Add internal peripheral

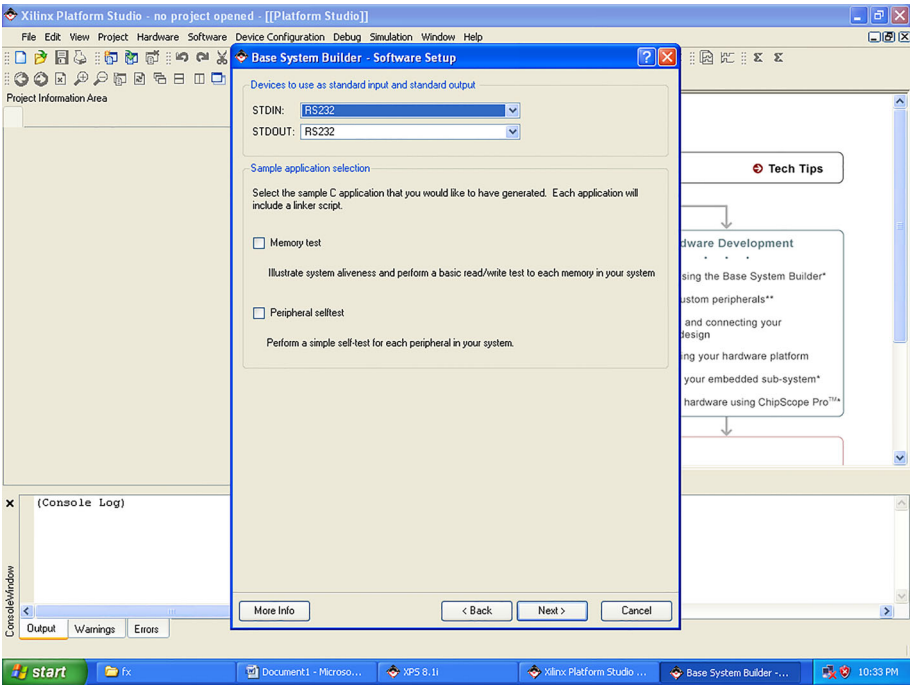

Fig. 18 Select interface cable

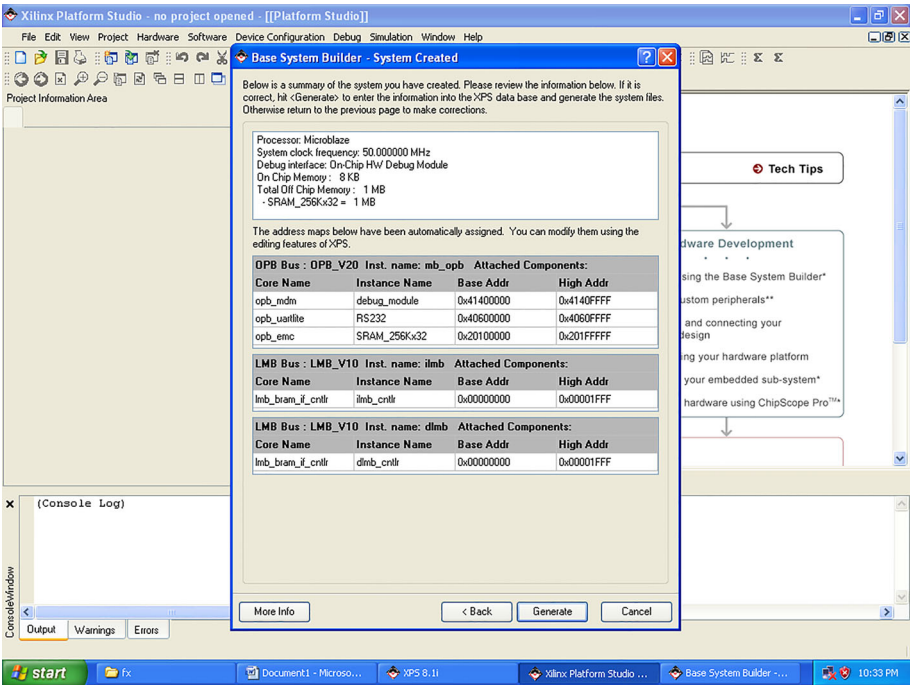

Fig. 19 System creating configuration

additional I/O Interface for the new data sheet and also we can select the peripheral device. Figure 17 shows the additional internal peripheral windows for the configuration. Figure 18 shows the selecting interface cable for the interconnection of the modules for the purpose of measuring the parameters value and also select the system configuration setup values. Figure 19 shows the system creating configuration. This section shows the summary of the XPS data base and on chip and off chip recollection limitation standards and also we can measure the system clock frequency. Figure 20 shows the Base system builder successfully created window then finish the base system builder. Figure 21 shows the start using of Xilinx Platform Studio. Here with, we have interfaced a new launching of Xilinx EDK 8.1.02 and start using the next platform studio.

Figure 22 shows the bit stream generation is created and add software tools box for the addition of the application to the project. Finally the bit stream generation is completed with saving the file in 'system.bit'. Figure 23 shows the add software application to the projects. In this module is used to give an input file for any type of images and then select the processor application. Figure 24 shows the system assembly view for add new file. After adding of new file, execute the micro blaze system. Herewith, we can measure the system assembly view of all the details.

Figure 25 shows the set of compiler options to all the configuration values and the environment details of executable mode and also select the debug and optimization details. In this, we can select the saving path and executable address path and also measure the output path. Figure 26 shows the generate linker script connection and to add pixel value of the entire pixel value of header file. It adds the pixel value of each and every size of the

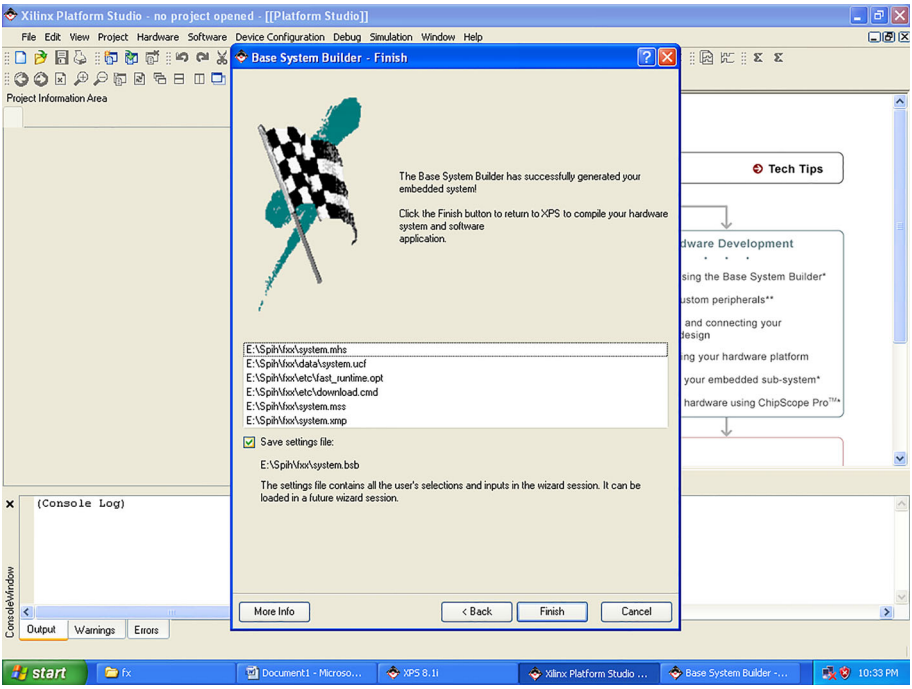

Fig. 20 Base system builder created

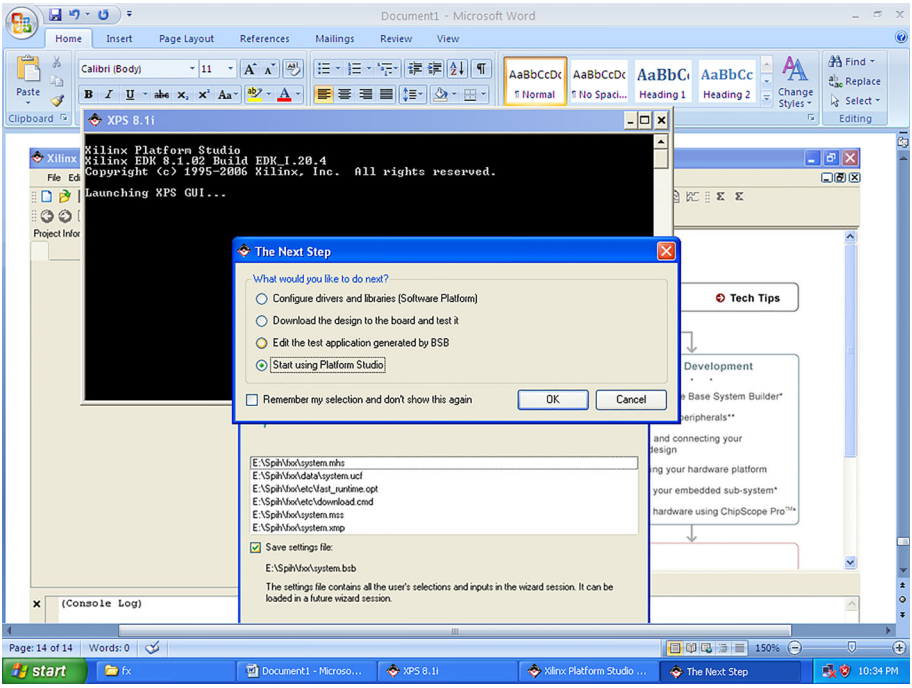

Fig. 21 Start using Platform studio

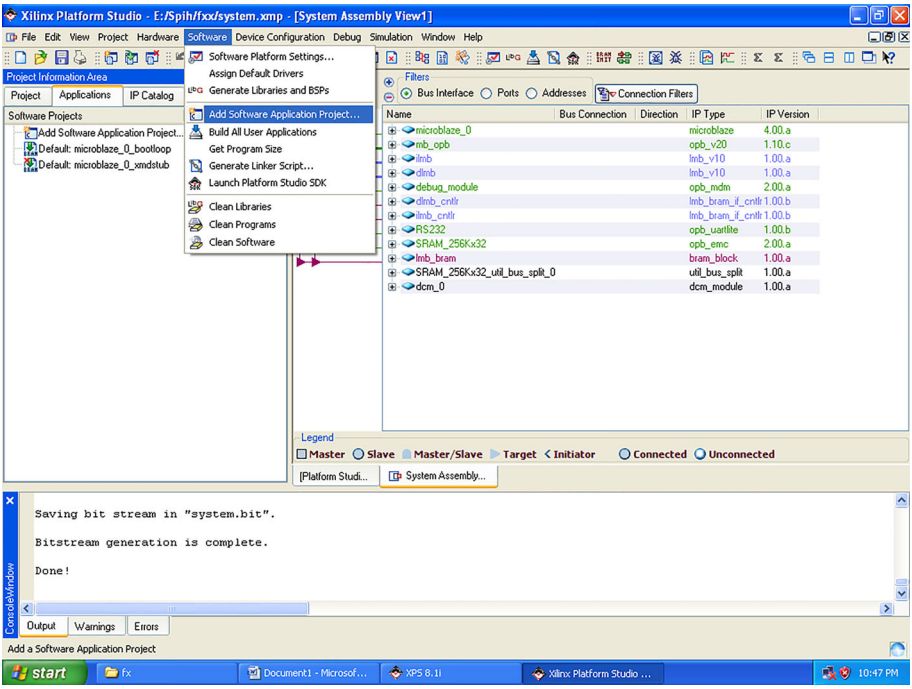

Fig. 22 Bit stream generation is created and add software application to the project

image and memory address are added and measured the entire values. We can also select the output executable link here.

Figure 27 shows the system assembly view which shows the bus interfaces and their connections. Here with connect the XMD file, After downloading bit stream, now we download application using XMD debugger in XPS. Now the FPGA board is ready to communicate on RS 232 cable link and then downloaded the programme successfully, the Elapsed time is 5 s.

Figure 28 shows the command window that is used to dump the program into the FPGA kit. Here the command file is saved as xmd.exe and we can see all details of the configuration data are measured in this window. Figure 29 shows the microblaze configuration of command window details and executable project image and processor configuration details are measured (Fig. 30).

In Table 1 shows the Specification table for the Spartan 3 EDK simulation of the entire step involved in the further processing. In this we have clearly given the entire specification, Operating frequency, processing time, Registers, supply details, using devices & parameters and then finally all the measured values are indicated. Table 2 describes the design summary of identification IC unit of various parameters used and available and the utilized percentages are listed. The performance of the proposed image compression approach is analyzed using the Image processing parameters in Table 3 and VLSI Parameters in Table 4.

In the image processing parameter we analysis the various parameter such as compression algorithm, image size, CR, BPP, MSE and pixel block size are analysed. In the analysis of VLSI parameter such as gate clock cycles required, power, processing rate, processing time, Array Size and Processor Area are analysed. When the image size is

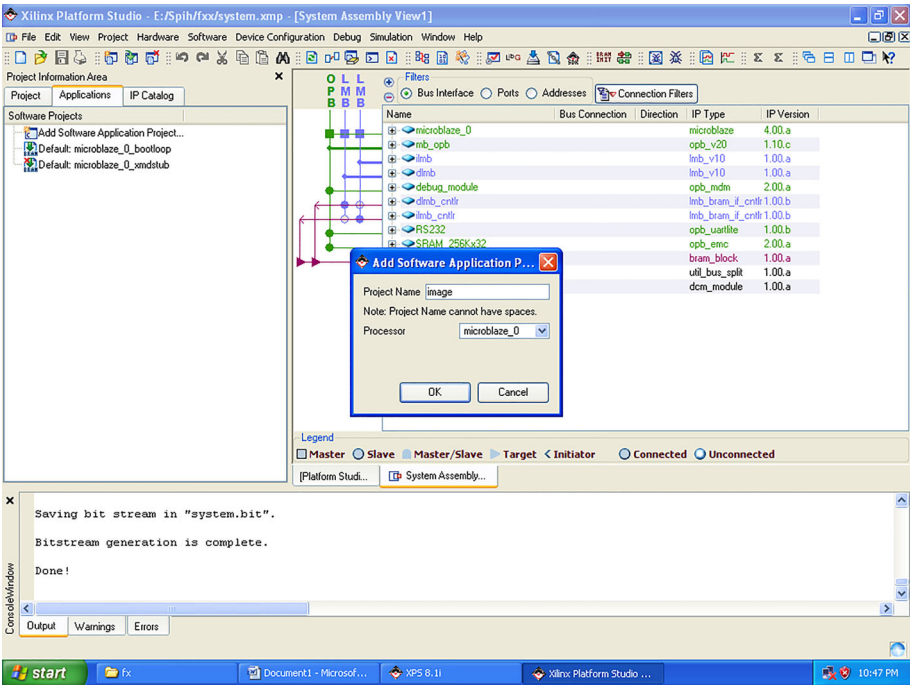

Fig. 23 Add software application

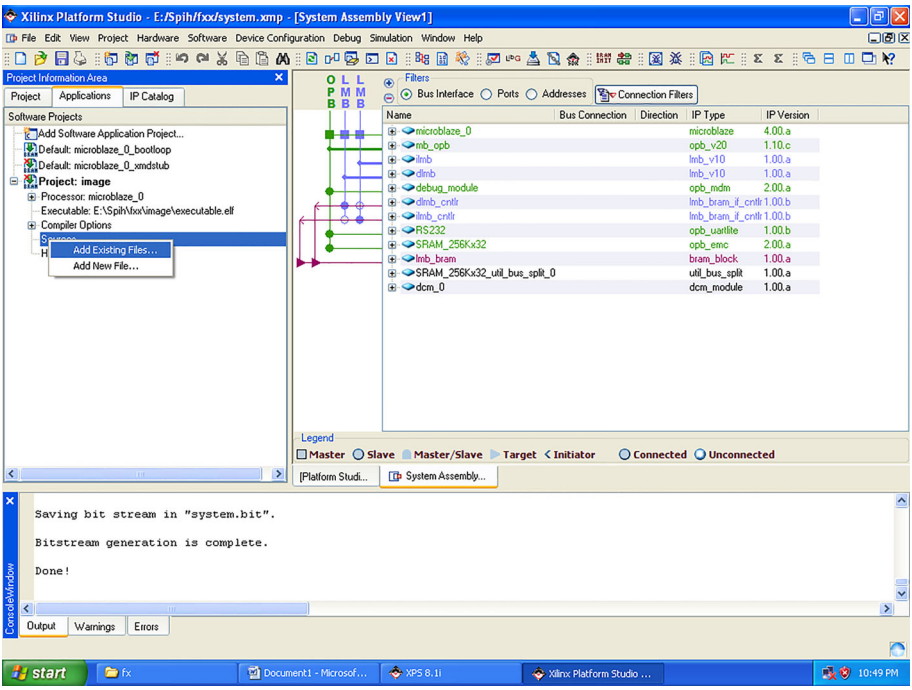

Fig. 24 System assembly view for add new file

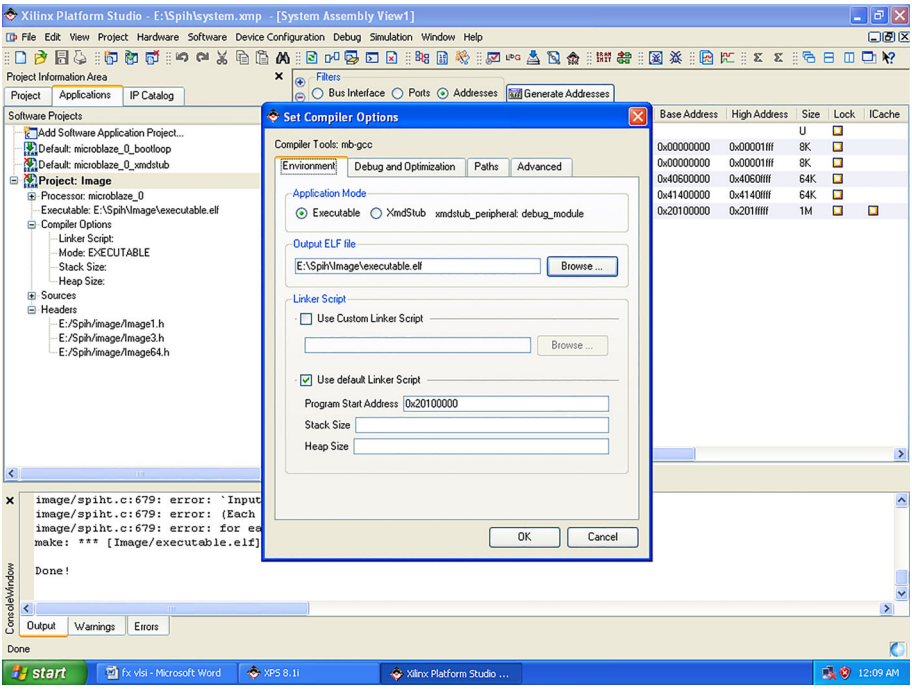

Fig. 25 To set compiler option

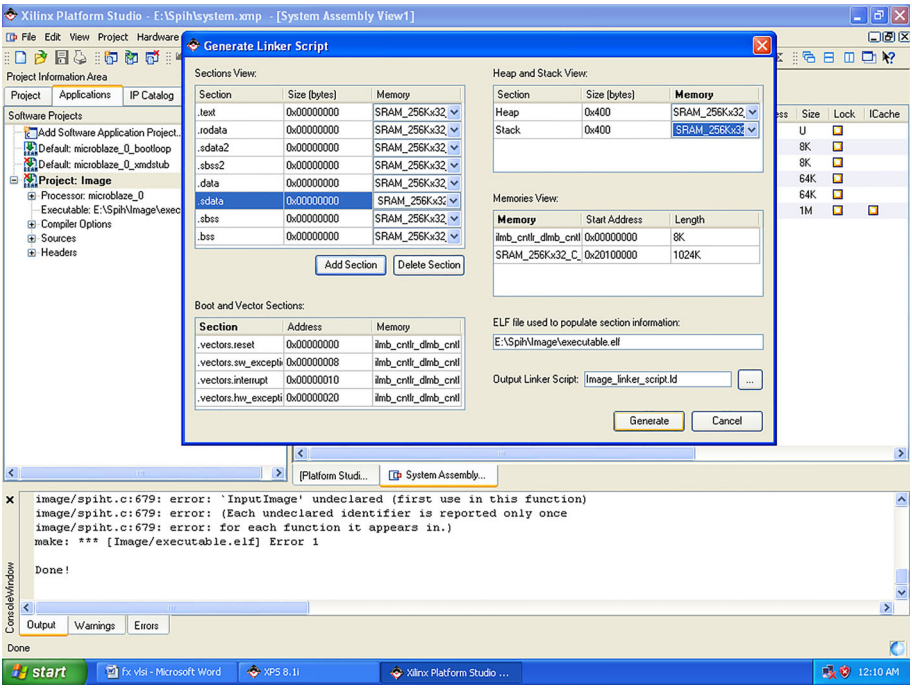

Fig. 26 Generate linker script and to add pixle value

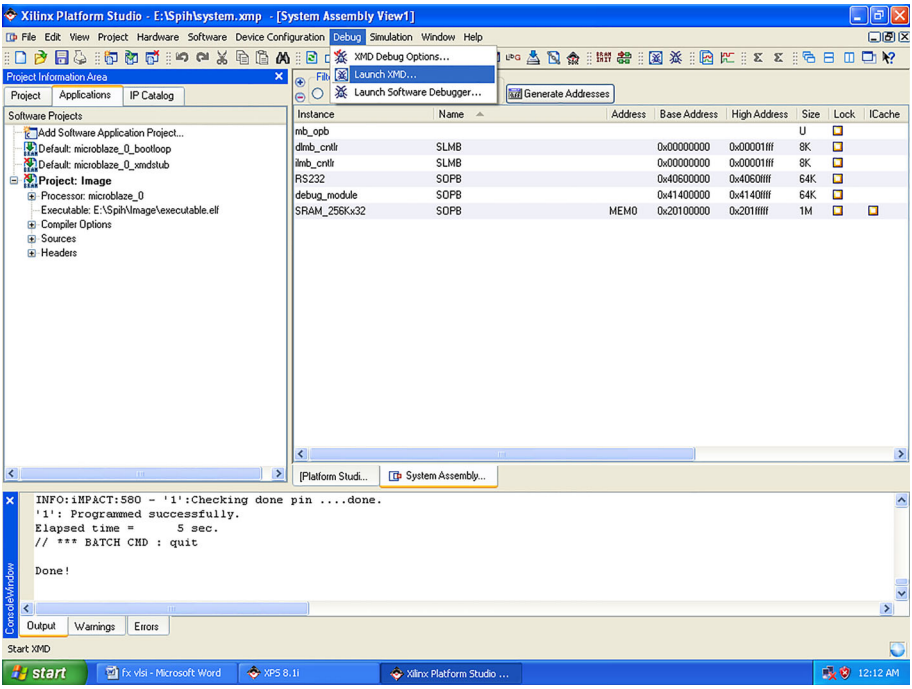

Fig. 27 System assembly view

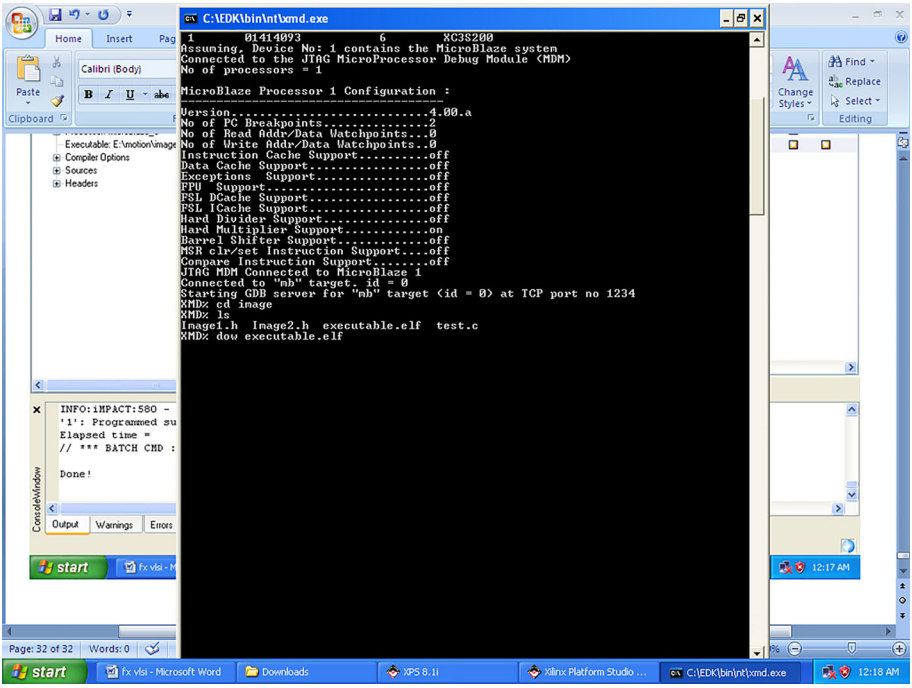

Fig. 28 Command window

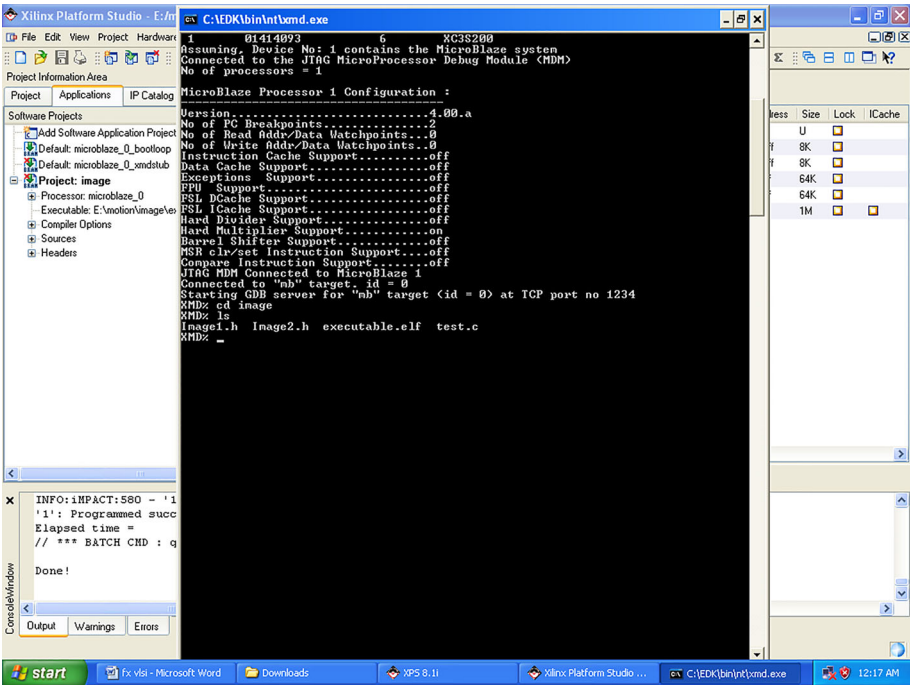

Fig. 29 Microblaze configuration of command window

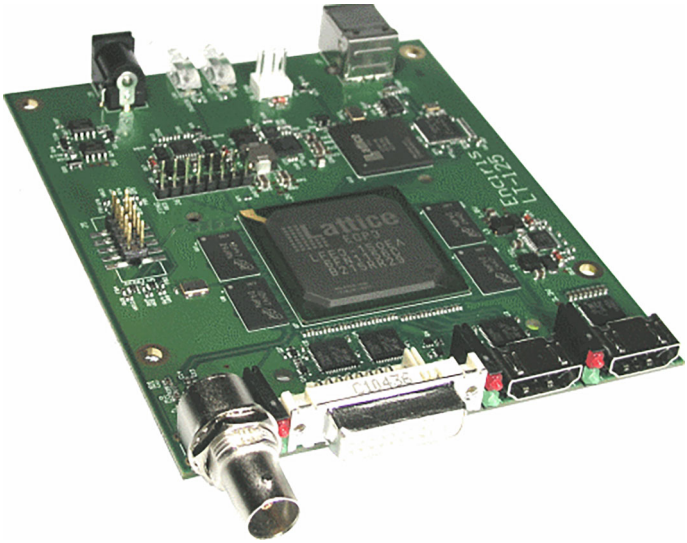

Fig. 30 Hardware module

Table 1 Specification table

Sl. No	Contents	Specification
1	Processor	Spartan 3e kit
2	Power supply	12 V
3	Current	1-1.3A
4	Serial port	UART
5	Parallel port	JTAG
6	Image pixel size	640 x 480
7	Re size image pixel	10 x 10
8	Clock frequency	33.739 MHz
9	Minimum period	29.639 ns
10	On chip memory	8 KB
11	Off chip memory	1 MB

Table 2 Design summary of identification IC unit

Particulars	Available	Used	Utilization (%)
Number of 4 input LUTs	28,672	4037	14
Number of occupied slices	14,336	3274	22
Number of bonded I/Os	484	81	14
Flip-flops	28,672	5520	19
Number of multi I/Os	16	1	6
Number of baud rate bit	9600	8	1
Number of global clocks	8	1	11

Table 3 Analysis of image processing parameter

Sl. No	Parameter	Range/values
1	Image size	640 × 480
2	Compressed image size	10 × 10
3	Compression type	Lossless
4	Compression algorithm	SPIHT Algorithm
5	Pixel block size	10 × 10
6	Compression ratio	1.3689
7	Bit per pixel	1
8	PSNR values	Infinity
9	Mean square error	0

changed to larger value, every module takes more time to complete the process. In the same concept we can analysis the different size of image takes different time as the unit of Seconds. The number of clock cycles is varied with respect to the size of the input images. When the size of the input image is larger, the approach needs more number of clock cycles to execute the process. But the power is not varied corresponding to images or image sizes. Similarly, the processing rate of every module provides the similar results.

Table 4 Analysis of VLSI parameter

Sl. No	Parameter	Range/Values
1	Number of processing time in Seconds	29.639 ns
2	Performance in term of clock cycles	41,94,304
3	Performance in term of power	328 μW
4	Performance in term of processing rate	11 cycles/pixels
5	Performance in term of logic gates	2121
6	Technology	0.5 μm
7	Array size	512 × 512
8	Processor area	0.36 mm^2
9	Number of baud rate bit	9600
10	Post processing Requirement	No
11	Maximum Frequency	33.739 MHz
12	Maximum output required time after clock	6.140 ns
13	Maximum combinational path delay	No path found
14	IOB Flip Flops	13

5 Conclusion and Future Work

The fast efficient SPIHT algorithm has been presented that operates through SPIHT and accomplishes completely with VLSI domain coding. The realization of this principle matched coding and decoding algorithm is new one and is shown to be more effective than in precious implementation of SPIHT code compression. Thus the pipeline architecture is used to makes itself a benchmark technique for its high effectiveness. The results of this coding algorithm with its fast execution are so impressive that is used for standardization in future image compression system. Finally to the best of our awareness, the fast efficient SPIHT coding algorithm is the first proposed implementation scheme in image processing. In terms of processing speed and memory it achieves very high performance.

Further improvements of the proposed system will reduce the computational complexity and reduce the image size using some other image compression technique and implement in kit level.

Acknowledgments This work was supported in part by Anna University Recognized Research Centre Lab at Francis Xavier Engineering College, Tirunelveli. We extend our gratitude to Dr. R. Ravi for his support and guidance. Also, we would like to thank the anonymous reviewers for their valuable comments and suggestions.

References

1. Muthukumaran, N., & Ravi, R. (2015). The performance analysis of fast efficient lossless satellite image compression and decompression for wavelet based algorithm. *Wireless Personal Communications, 81*(2), 839–859.
2. Muthukumaran, N., & Ravi, R. (2015). The performance analysis of VLSI based image acquisition using block based fast efficient compression algorithm. *International Arab Journal of Information Technology, 12*(4), 333–339.

3. Muthukumaran, N., & Ravi, R. (2014). Simulation based VLSI implementation of fast efficient lossless image compression system using Adjusted Binary Code & Golumb Rice Code. *World Academy of Science, Engineering and Technology, 8*(9), 1603–1606.
4. Akter, M., Reaz, M. B. I., Mohd Yasin, F., & Choong, F. (2008). A modified set partitioning in hierarchical trees algorithm for real compression. *Journal of Communication Technology Electronics, 53*(6), 642–650.
5. Ansari, M. A., & Ananda, R. S. (2009). Context based medical image compression for ultrasound images with contextual set partitioning in hierarchical trees algorithm. *Advance Engineering Software, 40*(7), 487–496.
6. Andra, K., Acharya, T., & Chakrabarti, C. (2000). A multi-bit binary arithmetic coding technique. In Proceedings of international conference image process, Vancouver, BC, Canada, vol. 1, pp. 928–931.
7. Cao, B., Li, Y. S., & Liu, K. (2004). VLSI architecture of MQ encoder in JPEG 2000. *Journal of Xidian Xuebao, 31*(5), 714–718.
8. Corsonello, P., Perri, S., Zicari, P., & Cocorullob, G. (2005). Microprocessor based FPGA implementation of SPIHT image compression subsystems. *Microprocessor and Microsystems, 29*(6), 299–305.
9. Shah, Devangkumar, & Vithlani, Chandresh. (2014). VLSI oriented lossy image compression approach using dA-based 2D-discrete wavelet. *International Arab Journal of Information Technology, 11*(1), 59–68.
10. Fry, T. W., & Hauck, S. A. (2005). SPIHT image compression on FPGAs. *IEEE Transactions on Circuits System for Video Technology, 15*(9), 1138–1147.
11. Liu, Kai, Belyaev, Evgeniy, & Guo, Jie. (2012). VLSI architecture of arithmetic coder used in SPIHT. *IEEE Transactions on Very Large Scale Integration Systems, 20*(4), 697–710.
12. Marpe, D., Schwarz, H., & Wiegand, T. (2003). Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits Systems for Video Technology, 13*(7), 620–636.
13. Palanisamy, G., & Samukutti, A. (2008). Medical image compression using a novel embedded set partitioning significant and zero block coding. *The International Arab Journal of Information Technology, 5*(2), 132–139.
14. Pan, H., Siu, W. C., & Law, N. F. (2008). A fast and low memory image coding algorithm based on lifting wavelet transform and modified SPIHT. *Signal Processing Image Communication, 23*(3), 146–161.
15. Said, A., & Pearlman, W. A. (1996). A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits & Systems for Video Technology, 6*, 243–250.
16. Wiseman, Y. (2001). A pipeline chip for quasi arithmetic coding. *IEICE Transactions on Fundamentals, 84*(4), 1034–1041.

Dr. N. Muthukumaran was born in Kaniyakumari, Tamilnadu, India, in 1984. He received the B.E (ECE) from Anna University, Chennai, India, in 2007 and the M.E (Applied Electronics) from Anna University, Chennai, India, in 2010. Where he is currently working toward the Ph.D. (Information and Communication Engineering) Degree, Anna University, Chennai, India. He is currently working as an Associate Professor and Research centre lab in Francis Xavier Engineering College, Tirunelveli. His major research interests are Image Processing/Compression, Digital and Analog Very Large-Scale Integration circuit design. He conducted several projects in the area of Image processing, Image Compression, Very Large-Scale Integration circuit. Since 2008 he has published more than 19 journals in International and 49 National/International conferences papers.

Dr. R. Ravi is an Editor in International Journal of Security and its Applications (South Korea). He is presently working as a Professor and Head of the Department of CSE, Francis Xavier Engineering College, Tirunelveli. He completed his B.E in Computer Science and Engineering from Thiagarajar College of engineering, Madurai in 1994 and M.E in CSE from Jadavpur Government research University, Kolkatta. He has completed his Ph.D. in Networks from Anna University Chennai. He has 18 years of experience. He is the first person in India from a private institution who was the judge (Chair person) for an International conference in IIT, Chennai. He published 43 International/National Journals, and 4 international Journals are under process. He actively participated in 23 international Conference, 98 National Conferences and bagged shields in many. He is also a full time recognized guide for various Universities. Currently he is guiding nine research scholars. His areas of interest are Virtual Private networks, Image Processing, Neural Network.