# Design and Implementation of Arithmetic Coder Used in SPIHT

**K. Harika, K. V. Ramana Reddy**

***Abstract*— In this paper Set Partitioning in Hierarchical Trees (SPIHT) algorithm for image compression is proposed with a arithmetic coder thereby it compresses the Discrete Wavelet Transform decomposed images. This architecture is advantageous from various optimizations performed at different levels of arithmetic coding from higher algorithm abstraction to lower circuit implementation. SPIHT has straightforward coding procedure and requires no tables which make a SPIHT algorithm an appropriate one for low cost hardware implementation. In order to avoid rescanning the wavelet transformed coefficients a breadth first search SPIHT without lists is used instead of SPIHT with lists. With the help of Breadth First search high speed architecture is achieved. Dedicated circuit such as common bit detector is used for loop unrolling the renormalization stage of arithmetic coding. Critical path in the architecture are shortened by employing Floating point multiplier and carry look ahead adder. Design has been implemented on Spartan 6 FPGA.***

***Index Terms*— Arithmetic coding, Common bit detection (CBD) circuit, Discrete wavelet transform (DWT), Set Partitioning in Hierarchical Trees (SPIHT).***

## I. INTRODUCTION

Image compression is an application of image processing performed on digital images. The main objective of image compression is to reduce redundancy of the image data in order to be able to store or transmit data in an efficient form. Uncompressed multimedia (audio and video) data requiresconsiderable storage capacity and transmission band width.[2] The main goal of compression is to provide sufficient storage space, wider transmission bandwidth and a longer transmission time for image, audio and video data. At present state of technology, the only solution to compress multimedia data before its storage and transmission and decompress it at the receiver. A Common characteristic of images is that the neighboring pixels are correlated and therefore contain redundant information. Two fundamental components of compression are redundancy and irrelevancy reduction. Redundancy reduction aims at removing duplication from the signal source (image/video).Irrelevance reduction omits only those parts of the signal that are not noticed by the signal receiver. In general, three types of redundancies that can be identified namely: Spatial Redundancy or correlation between neighboring pixel values. Spectral redundancy or correlation between different color-planes or spectral-bands. Temporal redundancy or correlation between adjacent frames in a sequence of images especially in video applications. Compression techniques are classified into two namely lossy compression and lossless compression.

**Manuscript received on August, 2013.**

**K.Harika**, VLSI Design and Embedded Systems, Visveswaraiah Technological University/ VTU Extension Centre, UTL Technologies Ltd. / Bangalore, India.

**Asst.Prof K.V.Ramana Reddy**, VLSI Design and Embedded Systems, Visveswaraiah Technological University/ VTU Extension Centre, UTL Technologies Ltd. / Bangalore, India.

Lossless compression technique guarantees the exact duplication of input in compress/decompress cycle. This is mainly used in medical applications. Methods for lossless image compression are: Run-length encoding, DPCM and Predictive Coding and Entropy encoding. With Lossy compression technique higher levels of data reduction is possible but it results in less perfect reproduction of the original image. This is useful in applications such as broadcast television, videoconferencing, and facsimile transmission, in which a certain amount of error is acceptable trade-off for increased compression performance.. Methods for lossy image compression are: Reducing the color space to the most common colors in the image, chroma sub-sampling, transform coding, and fractal compression. [1]

Generally a compression system consists of encoder and decoder stages. Encoder consists of the mapper, quantizer and encoder. Mapper transforms the input image into a format by reducing the inter-pixel redundancies. Quantizing refers to a reduction of the precision of the floating point values of the wavelet transform. A symbol encoder further compresses the quantized values to give a better overall compression. It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream will be smaller than the input stream. An arithmetic coder or Huffman coder is used as symbol encoders. Decoder part of the compression model consists of symbol decoder and inverse mapper. Among these FPGAs are best suited to implement image processing algorithms because they offers features like infinite reprogram ability, high parallelism, flexibility, computational power and short development time.[3,4].

## II. BACKGROUND AND MOTIVATION

### A. Discrete Wavelet Transform

A Discrete Wavelet Transform is a wavelet transform in which wavelets are discretely sampled.If Discrete Wavelet Transform is compared with the Fourier Transform it has an advantage of temporal resolution which captures both frequency and location information. A wave is an oscillating function of time or space and is periodic. In contrast, wavelets are localized waves.Wavelets have their energy concentrated in time and they are used for analysis of transient signals.

Wavelet transform uses wavelets of finite energy whereas the Fourier Transform and short Time Fourier Transform uses wavelets of finite energy.

A line based Wavelet Lifting scheme is used which mainy consists of three stages namely Split,Predict and Update Stages respectively. Figure 2.1 shows the diagram for Forward transform for lifting scheme.[6]

Split:This step is often refered to as Lazy Wavelet transform.In this stage the input pixel values obtained from input image 'X' is divided into even samples $S_{i+1}=X_e=\{x_{2j}\}$and odd samples $D_{i+1}=X_o=\{x_{2j+1}\}$, where 'i' represents the element and 'j' represents the iteration.

Predict:This step is often referred to as Dual lifting.In this stage the odd elements of the next iteration are predicted from the even samples of the present iteration.where 'i' represents the element and 'j' represents the iteration.

$$D_{i+1}=D_{i+1}-P(S_{i+1}) \quad (1)$$

Update:This step is referred to as Primary lifting. In this stage the even elements of next iteration are calculated from the odd samples of the update stage.Update stage follows the predict stage.

$$S_{i+1}=S_{i+1}+U(D_{i+1}) \quad (2)$$

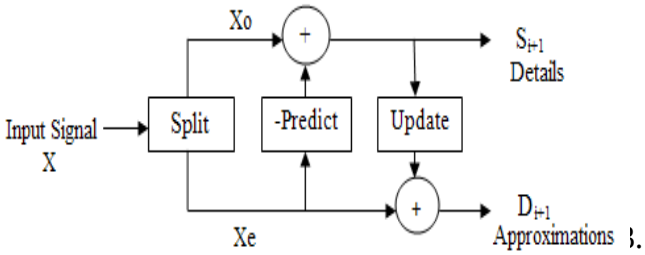


**Figure 2.1: Block diagram of Forward Lifting Scheme of DWT**

### A. Set Partitioning In Hierarchical Trees Algorithm

SPIHT is an image compression algorithm with lists to store the significant information of wavelet coefficients for image coding purpose.It mainly uses three different lists namely List of Significant Pixels(LSP),List of Insignificant sets(LIS) and list of Insignificant pixels(LIP).In SPIHTalgorithm initially all wavelet coefficients are considered as insignificant sets and these sets are placed in the list called LIS.Wavelet coefficients in LIS are compared with a predefined threshold if the wavelet coefficient value is greater than the predefined threshold then they are placed in the list called LSP and if the value is lesser than the predefined threshold then the wavelet coefficients are placed in the list called LIP.All the pixel values in the list LIS are tested for there significant state.Significant state of the wavelet coefficints are tested using the equation (3) which is as given below.So,this the reason why SPIHT algorithm less bits to code after the discrete wavelet transform. [4]

$$S_n(\tau)=\begin{cases}1 & max_{i,j\in\tau}\{C_{i,j}\}\geq 2^n \\ 0 & Otherwise\end{cases} \quad (3)$$

Where $C_{i,j}$ represents the coefficient value for the position i,j in the wavelet sub-band. Γ represents set of coefficients and Sn(Γ) represents the significant state of the set.

### B. Arithmetic Coding

Arithmetic coding is a variable-length source encoding technique. In traditional entropy encoding techniques such as Huffman coding, each input symbol in a message is substituted by a specific code specified by an integer number of bits. Arithmetic coding deviates from this paradigm. In arithmetic coding a sequence of input symbols is represented by an interval of real numbers between 0.0 and 1.0. The longer the message, the smaller the interval to represent the message becomes. More probable symbols reduce the interval less than the less probable symbols and hence add fewer bits in the encoded message. As a result, the coding result can reach to Shannon's entropy limit for a sufficiently large sequence of input symbols as long as the statistics are accurate. In arithmetic coding we make use of three registers namely low, high and range. Cumulative frequency is the defined as the cumulative counts of the symbol 'i' [7]. If current interval is given by (low, high) then the values of range, low and high are calculated using the formula as given in equation (4).

$$\begin{aligned} range &= high - low + 1 \\ high &= low + range * \frac{cum\_freq\ [i-1]}{cum\_freq\ [0]} \\ low &= high + range * \frac{cum\_freq\ [i]}{cum\_freq\ [0]} \end{aligned} \quad (4)$$

Where cum_freq[i] represents the cumulative frequency of the symbol 'i'.

For avoiding the underflowing of registers and to reduce the coding latency a normalisation procedure is used which is as follows:

1. If high < HALF, where HALF=0.5*$2^v$ then a '0' bit is written into output bit-stream.
2. If low ≥ HALF, then a '1' bit is written into output bit-stream.

Otherwise, an output bit is not defined. In this case a bit_to_follow counter is increased. Then if condition 1 is satisfied then a '0' bit and bit_to_follows ones are written into output bit-stream. If condition 2 is satisfied then a'1'bit and bit_to_follows zeros are written into output bit-stream.

## III. DESIGN AND IMPLEMENTATION

The block diagram of the SPIHT encoding is as shown in the Figure 3.1. The input image is gray scale image of size 512x512 each pixel of 8 bits.

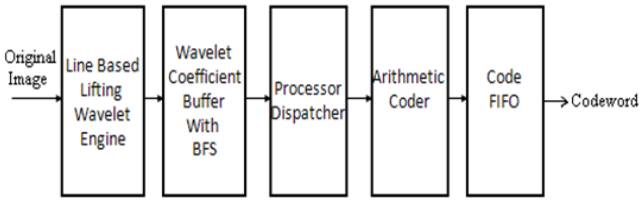


**Figure 3.1: Block diagram of SPIHT Encoding**

Input image is converted to pixel values in MATLAB.Pixel values obtained are given as input to the Line based wavelet lifting module which produces wavelet coefficients.

The transformed wavelet coefficients are placed in a buffer and they are accessed in a SPIHT-Breadth First Search way.Processor dispatcher dispatches the transformed wavelet coefficients to the arithmetic coder through internal bus. Output of arithmetic coder is provided to the internal bus and sent to the code FIFO. The Read FIFO and Truncate module are responsible for the final code stream formation, which reads each code FIFO from top to bottom and truncates the code stream according to the bit rate requirement.

### A. Line Based Lifting Wavelet Engine

Lifting step scheme consists of three simple phases: the first step, or Lazy wavelet, splits the data into two subsets: even and odd;

the second step calculates the wavelet coefficients (high-pass) as the failure to predict the odd set based on the even set; finally the third step updates the even
set using the wavelet coefficients to compute the scaling function coefficients (low-pass). The predict phase ensures polynomial cancellation in the high-pass, and the update phase ensures preservation of moments in the low-pass.

In folded architecture the output of processing element is fed back through the delay registers. By adding different number of delay registers and coefficients of processing elements folded architecture based lifting scheme can be used to design different wavelets. Folded architecture for (9, 7) wavelet with line based lifting wavelet engine is as shown in Figure 3.2.Coefficients for (9, 7) are 0.5 and 0.25which are represented by a, b, c, d. Two delay registers (D) are used in order to schedule the data in each phase. Based on the phase of interleaved computation, the coefficient for multiplier M1is either a or c or similarly the coefficient for multiplier M2 is b or d. Critical path in this folded architecture is reduced. [6]

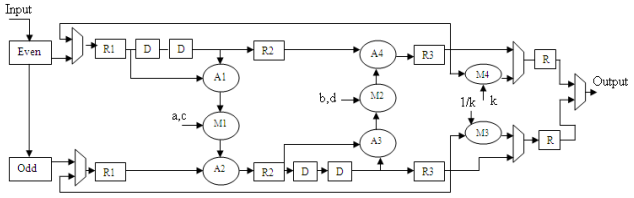


**Figure 3.2: Architecture of Lifting based Wavelet Engine**

Figure 3.3 illustrates the steps for performing a two-level DWT on an image. The 1-D DWT is first performed on the rows of the image producing low-frequency $L_1$ and high-frequency components $H_1$. After performing a 1-D DWT again on the columns of and the first level of decomposition is completed $LL_1$, $LH_1$, $HL_1$ and $HH_1$ are obtained.

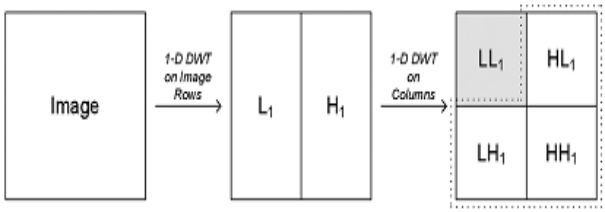


**Figure 3.3: Applying DWT on an image**

## B. SPIHT Algorithm

SPIHT is the wavelet based image compression method. It provides the Highest Image Quality, Progressive image transmission, Simple quantization algorithm, Lossless compression, exact bit rate coding and Error protection. SPIHT makes use of three lists – the List of Significant Pixels (LSP), List of Insignificant Pixels (LIP) and List of Insignificant Sets (LIS). These are coefficient location lists that contain their coordinates. After the initialization, the algorithm takes two stages for each level of threshold – the sorting pass (in which lists are organized) and the refinement pass. The result is in the form of a bit stream. It is capable of recovering the image perfectly by coding all bits of the transform. Figure 3.4 shows the flow chart of SPIHT algorithm.

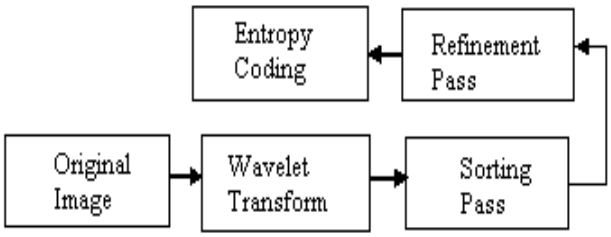


**Figure 3.4: Flow Chart of SPIHT algorithm**

In order to avoid the multiple scan of wavelet coefficients, this is difficult for real time hardware implementation scanning of wavelet coefficients is carried out in a Breadth First search way. The SPIHT-BFS visits each coefficient only once and outputs coding information.

## C. Architecture of Arithmetic Coder

The architecture of arithmetic coder is as shown in Figure 3.5.

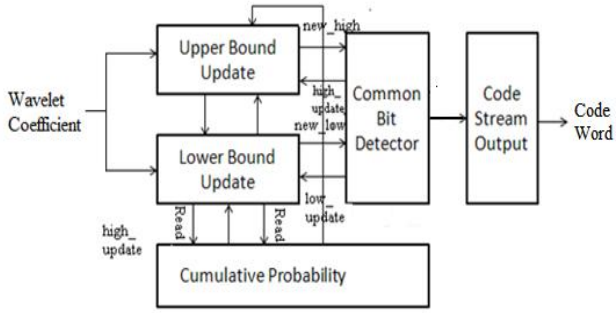


**Figure 3.5: Architecture of Arithmetic Coder**

The arithmetic coder consists of three main parts, upper and lower bound update, and common bit detector. The tree construction part visits the wavelet coefficients by the breadth first search order. For speedup, all valid bit-planes are scanned in parallel. Significant wavelet coefficients are provided as input to the arithmetic coder. Upper bound and lower bound update units in arithmetic coder are used for updating the initial intervals to 0 and 1.High and low values are multiplied with cumulative probabilities and updated high and low values are given as input to common bit detector unit. Output is then passed to the code stream output unit in arithmetic coder. Cumulative probability part will hold the frequency of occurrence of wavelet coefficient. Upper bound and lower bound update units in arithmetic coder are implemented with carry look ahead adder and floating point multiplier. [7]

## D. Upper Bound and Lower Bound Update

Upper bound and lower bound update registers consists of Look ahead adder and Fast array multiplier. The calculation units for upper and lower bound update units are as shown in the Figure 3.6.

Output of upper and lower update units are new_low and new_high. For speed up purpose, a Carry Look Ahead adder and a floating point multiplier are employed to reduce the delay of critical path. The cumulative probabilities cum_freq[i] and cum_freq [i-1] are provided by the cumulative probability module and this is accompanied by high and new values to compute new bounds for the probability interval these are calculated using the formula (4).

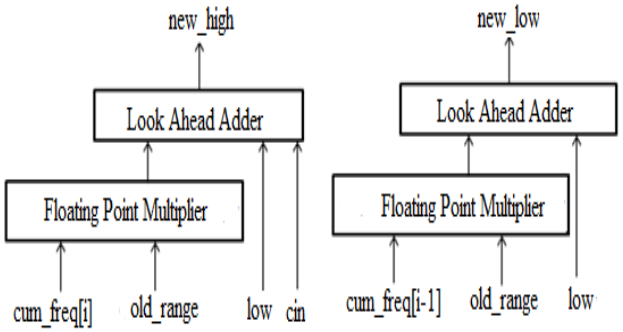


**Figure 3.6: Calculation units for upper and lower update unit**

### E. Common Bit Detector

The internal structure of common bit detector is as shown in the Figure 3.7. The new bound values are then registered and connected to the common bit detector (CBD) part which unrolls the internal loop and records the same bits from the MSB to the LSB between two registers. [7]

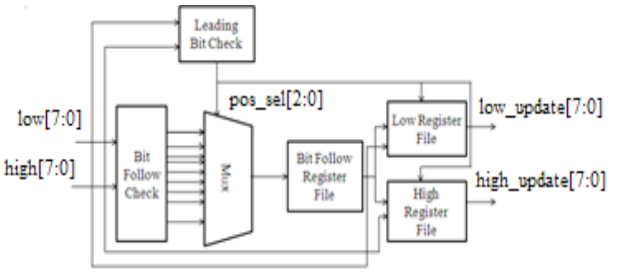


**Figure 3.7: Internal Structure of common Bit Detector**

Internal Structures of Leading bit check (LBC) and bit follow check (BFC) are as shown in the Figure 3.8.

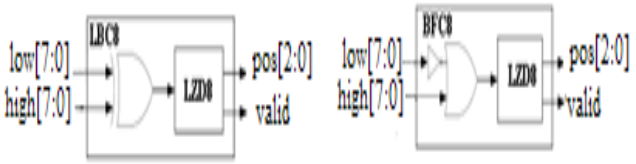


**Figure 3.8: Internal Structure of LBC and BFC**

LBC consists of XOR gate and leading zero detectors [4] which detects common bits between high and low registers.BFC module checks the underflow

## IV. TESTS AND RESULTS

The design of SPIHT algorithm with arithmetic coder is described in VHDL. Design and testing of individual module has been carried out. Figure 4.1 depicts the simulation results of lifting based DWT. The input to this module is the pixel values stored in 1-D array. Output of this module is a transformed coefficient.

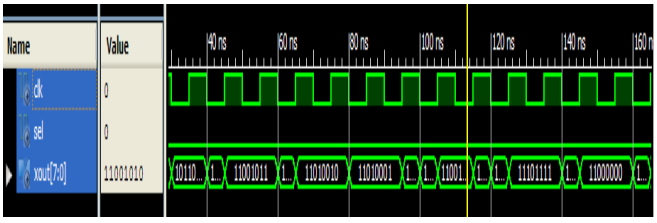


**Figure 4.1: Simulation result of lifting based DWT**

Figure 4.2 shows the simulation output of DWT in MATLAB software. Gray scale image of 512X512 is given as input.1-D DWT is first applied on columns and then on the rows.

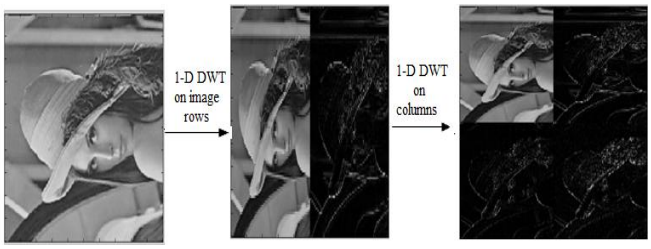


**Figure 4.2: Simulation result of lifting based DWT in MATLAB**

Output of DWT is provided as input to SPIHT algorithm so the transformed wavelet coefficients are initially stored in the List of Insignificant Sets (LIS) now the wavelet coefficients in the LIS are compared with the threshold value if the wavelet coefficients are greater than the threshold then they are placed in List of significant pixels otherwise they are placed in List of Insignificant Pixels.

Output of SPIHT-BFS algorithm is as shown in the Figure 4.3.

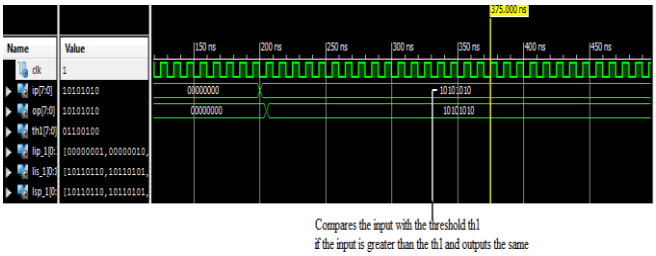


**Figure 4.3: Simulation result of SPIHT algorithm**

Figure 4.4 and 4.5 depicts the simulation result for upper bound update and lower bound update. Output of this module updated low and high values which are provided as input to common bit detector.

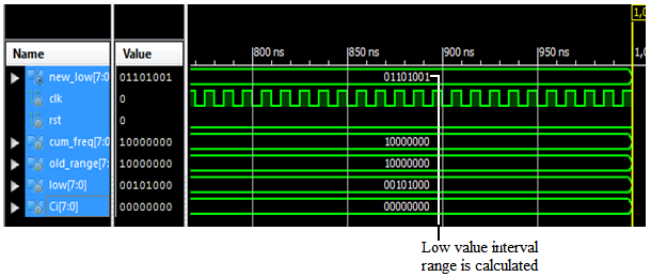


**Figure 4.4: Simulation result of Lower bound update module**

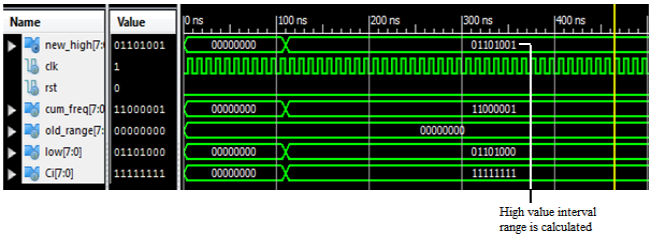


**Figure 4.5: Simulation result of Upper bound update module**

Figure 4.6 shows the simulation result of common bit detector module which unrolls the renormalization stage of arithmetic coder.

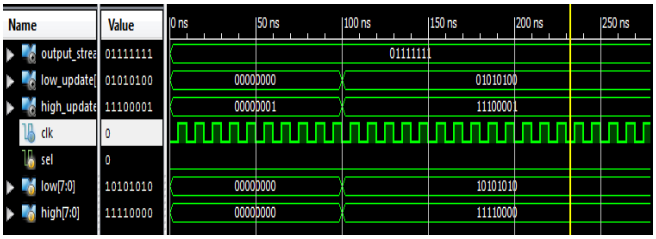


**Figure 4.6: Simulation result of common bit detector**

Figure 4.7 shows the simulation result of an arithmetic coder.

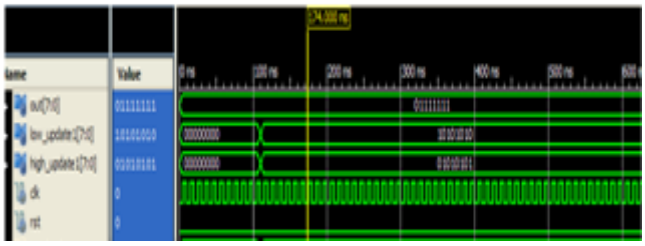

**Figure 4.7: Simulation result of Arithmetic Coder**

The Simulation result for SPIHT algorithm with arithmetic coder after integrating all the modules is as shown in Figure 4.8. The final output consists of codeword generated which is of 8bits.

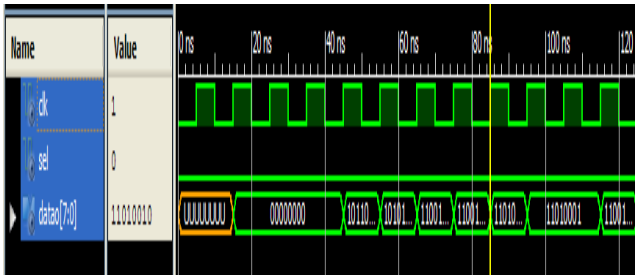


**Figure 4.8: Simulation result of SPIHT algorithm with arithmetic coder after integration.**

Figure 4.9 shows the simulation result for SPIHT algorithm with arithmetic coder after integration in ModelSim.

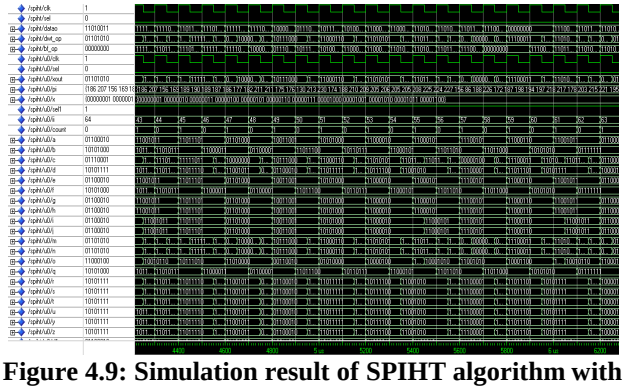

**Figure 4.9: Simulation result of SPIHT algorithm with arithmetic coder after integration in ModelSim**

Line based discrete wavelet engine performs the split, predict and update operations on the input pixels of an image to obtain the transformed wavelet coefficients at the output.

Figure 4.10 shows the Register Transfer Logic view of the discrete wavelet Transform applied on the input Pixel values of an image.

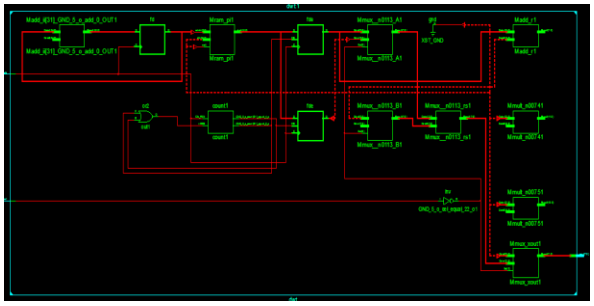

**Figure 4.10: RTL view of Discrete Wavelet Transform**

Figure 4.11 shows the Register Transfer Logic view of the SPIHT algorithm.

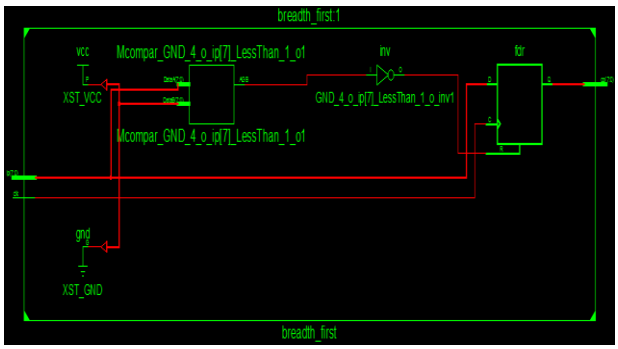


**Figure 4.11: RTL view of SPIHT algorithm**

Figure 4.12 shows the Register Transfer Logic view of an arithmetic Coder.

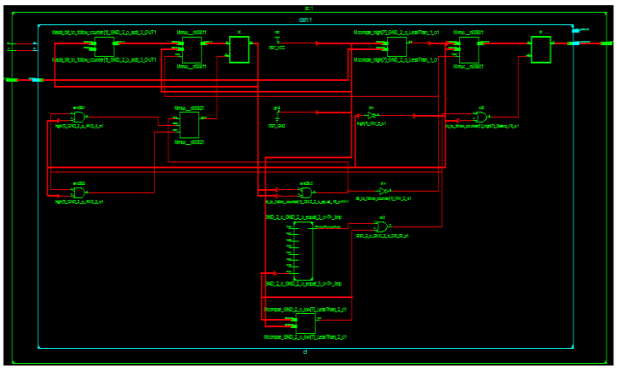

**Figure 4.12: RTL view of Arithmetic Coder**

RTL view of the entire design after integration is as shown in the Figure 4.13.

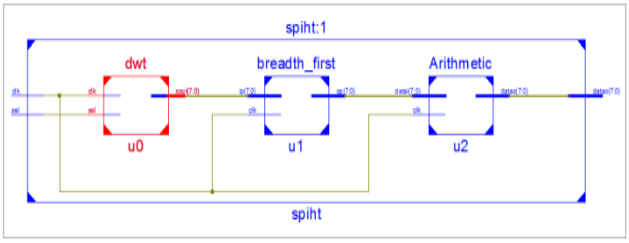


**Figure 4.13: RTL view of the entire design**

Entire design is implemented on Spartan 6 FPGA and the design described in VHDL is synthesized and the output is observed on Chipscope Pro Window, the option available in Xilinx ISE 13.2 software for analyzing the hardware results Figure 4.14 shows output of the entire design obtained on Chipscope Pro Window.

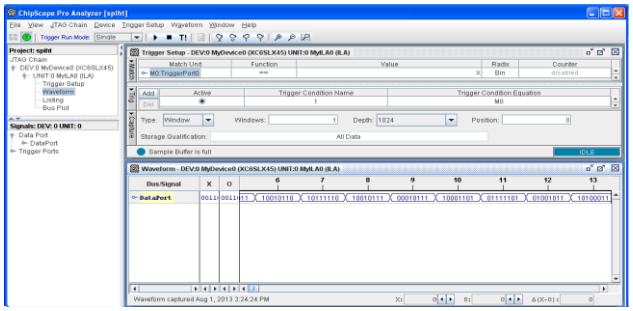

**Figure 4.14: Output obtained on Chipscope Pro window of Xilinx 13.2 ISE**

## V. CONCLUSION

This paper presents implementation of arithmetic coder used in SPIHT. The hardware is realized on Spartan 6 FPGA kit. Output of the entire design is observed on ChipScope Pro window in Xilinx ISE 13.2. A compression of 4:1 is achieved in this architecture so the memory occupied by Codeword is reduced by four times when compared to the input with a bit-rate of 8bpp. As a result smaller storage space is needed to store the encoded bit-stream and it is easy to transmit encoded bit-stream in lesser transmission bandwidth. For a pixel precision of 8 bits with a resolution of 512 x512, a throughput of coder is 800Mb/s.

For improvement of throughput purpose SPIHT algorithm without lists can be implemented. A simple context scheme can be employed.

## ACKNOWLEDGMENT

I acknowledge Dr. V. Venkateswarlu, Principal, VTU Extension Centre, UTL Technologies Ltd., Bangalore for his guidance and suggestion and UTL Technologies Ltd., Bangalore for providing lab facility during the design and implementation.

## REFERENCES

1. C. Chrysafis & A. Ortega, “Line based, reduced memory, wavelet image compression”,IEEE Trans. Image Process., Vol 9, No. 3, Sep.2000,pp.378–389.
2. Kai liu,Eygeniy,Belyaey and Jie Guo,”VLSI architecture of arithmetic coder used in SPIHT”,IEEE transactions on VLSI Systems, Vol 20,No.4,April 2012.
3. Rehna.V.J , Shubhangi.S & Vasanthi.S,”Improving the performance of Wavelet based image compression using spiht algorithm”, IRNet Transactions on E and E Engineering (ITEEE) Vol 1,Iss 2,2012.
4. V. G. Oklobdzija, “An algorithmic and novel design of a leading zero detector circuit: Comparison with logic synthesis,” IEEE Trans on VLSI systems, Vol. 2, No. 1, Mar. 1994, pp. 124–128.
5. I.C.Witten, R. M. Neal, and J. G. Cleary, “Arithmetic coding for data compression,” Commun. ACM, vol. 30, no. 6, Jun. 1987, pp.520-540.
6. Usha Bhanu.N and Dr.A.Chilambuchelvan,”A Detailed Survey on VLSI Architectures for Lifting based DWT for efficient hardware Implementation, VLSICS, Vol.3, No.2, April 2012.
7. K.SivaNagiReddy, V.Sidda Reddy, Dr.B.R.Vikram,” Efficient Memory and Low Complexity Image Compression Using DWT with Modified SPIHT Encoder”, International Journal of Scientific & Engineering Research, Vol 3, Issue 8, 2012.
8. Bibhuprasad Mohanty, Abhishek Singh & Sudipta Mahapatra,“A high performance modified SPIHT for scalable image compression”, International of Image processing (IJIP), Vol.5, 2011.
9. D. Taubman, “High performance scalable image compression with EBCOT”, IEEE Trans. Image Process. Vol. 9, No. 7, July 2000, pp. 1158–1170.
10. F.W.Wheeler and W.A.Pearlman,”SPIHT image compression without lists”,Proceedings of IEEE International Conference on Acoustics, Speech and signal processing, ICASSP,Vol-4,June 2000,pp.2047-2050