İSTANBUL UNIVERSITY
İ|C|E|R|R|A|H|P|A|Ş|A|

RESEARCH ARTICLE

Electrica 2019; 19(1): 29-36

Enhanced SPIHT Algorithm with Pipelined Datapath Architecture Design

Serap Çekli, Ali Akman

Department of Computer Engineering, Maltepe University, Istanbul, Turkey

Cite this article as: Çekli S, Akman A. Enhanced SPIHT Algorithm with Pipelined Datapath Architecture Design. *Electrica*, 2019; 19(1): 29-36.

ABSTRACT

Set partitioning in hierarchical trees (SPIHT) is an efficient algorithm which is used for the image compression widely. SPIHT operates sequentially so, its parallel implementation is difficult. In this study, the SPIHT algorithm is improved for providing that it is suitable for the parallel processing applications, and the corresponding pipelined datapath is designed for the proposed enhanced SPIHT algorithm. The datapath is designed to have three stages as preprocessing, list generation and output stream. In the preprocessing stage, the flags which are supports the list generation stage are constituted. List of insignificant sets (LIS), list of insignificant pixels (LIP) and list of significant pixels (LSP) are formed in list generation stage. These lists contain the bit values which generate the output bit stream. The performance of the improved datapath design has been tested by compressing different images, and the obtained results are given.

Keywords: SPIHT, enhanced SPIHT, image compression, pipelined datapath

Introduction

The images require large storage capacity and high transmission bandwidth resulting from including considerable information. The compression operation which provides to eliminate the redundant information is advantageous for many aspects such as storage capacity, transmission speed.

Therefore, several studies have been fulfilled for the image compression, and still much efforts continue to improve the compression performance from some different aspects. Some of these studies have been focused on wavelet transform which provides the multi resolution analysis, based compression algorithms such as embedded zero tree wavelet (EZW), set partitioning in hierarchical trees (SPIHT), optimal truncated embedded block coding (EBCOT) used in JPEG2000 standard [1-3].

Set partitioning in hierarchical trees is a discrete wavelet transform (DWT) based, efficient, adaptive and computationally fast compression algorithm [2]. This algorithm is known as an enhanced implementation of the EZW algorithm which has improved performance, which provides embedded coding output. The algorithm provides satisfactory PSNR (peak signal to noise ratio) values, effective image quality and compression ratio, and fast processing time. It can be used for the lossy or lossless data compression according to the necessary application. By means of the mentioned advantages, SPIHT is a suitable option for image compression applications especially for small devices with limited power consumption and memory capacity [4, 5].

Until today, some studies have been performed to increase the performance of the SPIHT coders, and many researches are carried on in this field yet. Several studies which have been fulfilled have aimed at providing improvement of the compression performance [6], reduction of the memory requirement [4, 5, 7, 8], decrease of the output coding time [9-12], to provide efficient hardware resource usage [13, 14], security of the SPIHT algorithm [15, 16] or biomedical applications [17].

Corresponding Author:
Serap Çekli

E-mail:
serapcekli@maltepe.edu.tr

Received: 03.08.2018

Accepted: 27.08.2018

DOI: 10.26650/electrica.2018.15101

The use of breadth first search (BFS) procedure is a suitable option while implementing the architectural design of SPIHT algorithm to examine the SOTs. At first, BFS deals with the examination of neighboring pixels in the same level. By this means, extra register necessity is eliminated when the inspection is carried out for the decision process on a tree, and also parallel processing architecture could be designed.

The rest of the study can be summarized as follows; The SPIHT algorithm and the system architecture are described after the underlying concepts of the SPIHT algorithm are given. Then, the obtained results have been presented and discussed. As the last, the conclusion section is given.

SPIHT algorithm

In the SPIHT algorithm, the image is divided into quadtree structure repeatedly by calculating the DWT coefficients. In the algorithm, the values or positions of the pixels in the subbands are not transmitted. Instead, the decision points that define the structural properties of the image in the tree structure and the results of the decisions at these points are coded as output. Since the positions of the decision points in the tree structure and the output values are known, the image can be recovered in the decoder.

The SPIHT algorithm is consists of three steps which are initialization, sorting pass, and refinement pass. The algorithm uses three types of list structures for memory operations: LIS; list of insignificant sets, LIP; list of insignificant pixels, and LSP; list of significant pixels. A threshold value is determined considering the maximum coefficient value at the initialization step. Significant coefficients in LIP and LIS are determined at the sorting pass. For a value of n, a C_{ij} coefficient is significant if satisfies the condition $|C_{ij}|\geq 2^n$, otherwise it is insignificant. The significance test is applied to coefficients by arranging as sets, and the sets are partitioned by spatial orientation trees (SOTs). The SOTs are given in Figure 1.

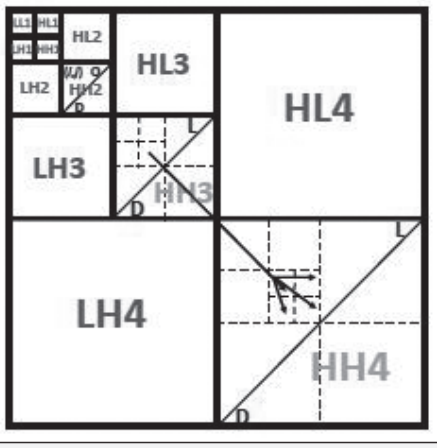

Figure 1. The spatial orientation trees of SPIHT algorithm, and subbands

Each pixel has four offspring except for the upper left pixel. $O(i,j)$: Defines the set of the coordinates of the four offspring of a pixel at the (i,j).

• $D(i,j)$: The set of all descendants of a position (i,j).

• $H(i,j)$: The set including the coordinates of the roots of all SOTs.

• $L(i,j)$: The set of the coordinates of all descendants excluding offspring of a position (i,j), and given as $D(i,j) - O(i,j)$.

The entries in the lists are determined by a (i,j) coordinate. These coordinates represent the individual pixels in LIP and LSP, and also an A type entry of the set $D(i,j)$ or a B type entry of the set $L(i,j)$.

The significance test applied to a given set of is defined in the following equation;

$$S_n(T) = \begin{cases} 1, & max_{(i,j)\in T}|c_{i,j}| \geq 2^n \\ 0, & \text{otherwise} \end{cases}, \quad (1)$$

Enhanced SPIHT algorithm

One of the main obstacles to design of the high performance digital system of the SPIHT compression algorithm is difficulty of the parallel implementation resulting from use of lists obtained in one step of algorithm in the next step. In this study, a novel algorithm which reconfigures the constitution of LIP and LIS is proposed to eliminate this drawback, and the outline of the proposed algorithm is given in the Figure 2. The algorithm performs list constitution based on BFS in quadtree structure.

In the step of construction of LIP and LIS, the constitution of LIP and LIS for the corresponding threshold value is fulfilled. The dependency between consecutive threshold values is removed because the constitution of these lists are repeated again for each new threshold value. Construction of LIP and LIS starts with four coefficients located at the lowest frequency band of wavelet transform coefficient matrix. Coefficient coordinates not in LSP are inserted in LIP. Other entries of LIP are obtained while the construction of LIS.

LIS is obtained by testing in sequence the roots of three subtrees. The root (i,j) of subtree is inserted to LIS. The offspring $(O(i,j))$ of that coordinate not in LSP is inserted to LIP, if the result of significance test of all descendants of (i,j) coordinate is $Sn(D(i,j))=1$. After that, significance test of all descendants excluding offspring of (i,j) coordinate is performed.

If $Sn(L(i,j))=1$ then (i, j) subtree is tested using BFS. In this test, if significance test of all descendant excluding offspring of the parent pixel of a (k,l) coordinate in subtree is $Sn(L(parent(k,l)))=1$, this (k,l) coordinate is added to LIS. If the significance test of all descendant of that (k,l) coordinate is $Sn(D(k,l))=1$, then offspring $(O(k,l))$ not in LSP are added to LIP. Thus, test continues on the other subtree upon completion of scanning entire subtree.

Step1. *Initialization:* $n = \lfloor log_2 max_{i,j}(c_{i,j}) \rfloor$ is transmitted. LSP is constructed as an empty list.

Step2. *Construction of LIP ve LIS:*

LIP and LIS lists are constructed as empty sets.

Each $(i,j) \in H$ and $(i,j) \notin$ LSP pixels are added to

For each $(i,j) \in H$

If $(i,j) \neq (0,0)$ then (i,j) is added to LIS.

If $S_n(D(i,j)) = 1$ then

Each $(k,l) \in O(i,j)$ and $(k,l) \notin$ LSP pixels are added to LIP.

If $S_n(L(i,j)) = 1$ then

BFS(i,j) set is constructed from set $D(i,j)$ by breadth first search.

For each $(k,l) \in$ BFS(i,j)

If $S_n(L(parent(k,l))) = 1$ then

(k,l) is added to LIS.

If $O(i,j) \neq \emptyset$ and $S_n(D(i,j)) = 1$

Each $(p,r) \in O(k,l)$ and $(p,r) \notin$ LSP pixels are added to LIP.

Step3. *LIS output stream:*

For each $(i,j) \in$ LIS

If $S_n(D(i,j)) = 1$ then a 1 otherwise a 0 is added to LIS output stream.

If $S_n(L(i,j)) = 1$ then a 1 otherwise a 0 is added to LIS output stream.

Step4. *LIP output stream and LSP set:*

For each $(i,j) \in$ LIP

If $S_n(i,j) = 1$ then

A 1 is added to LIP output stream.

The sign of $c_{i,j}$ is added to LIP output

(i,j) is adde to LSP.

If $S_n(i,j) = 0$ then a 0 is added to LIP output

Step5. *LSP output stream:* For each entry of (i,j) n-th most significant bit of $|c_{i,j}|$ is added to LSP output stream except for entries appended in last sorting pass (for the same n).

Step6. *Output stream:* LIS, LIP and LSP output streams are added to output stream.

Step7. *Quantization step update:* n is decreased 1 and if there is need the process continues from Step2.

Figure 2. Enhanced SPIHT coding algorithm

In LIS output stream, two bits are coded in for significance of all descendant and all descendant excluding offspring for each coefficient. Each coefficient in LIP is subjected to significance test. If the result of significance test for coefficient is 1, a 1 and sign of coefficient is added to LIP output stream. This coefficient coordinate is added to LSP. Otherwise, 0 is coded to LIP output stream. LIP output stream and LSP are completed by this manner. The LSP output stream is similar to the refinement pass in the original algorithm. The threshold value is halved and the process continues by the construction of the new LIP and LIS. The enhanced SPIHT coding algorithm is given in Figure 2.

Proposed SPIHT architecture

The architectural design of the enhanced SPIHT algorithm is described in this section. This architectural design has been performed in pipelined datapath form. The system flow diagram of the designed architecture is given in Figure 3.

The block diagram of the proposed SPIHT architecture is given in Figure 4. The architecture is designed in the form of a three stage pipelined architecture corresponding with the system flow in Figure 3. The first stage of the designed architecture is the preprocessing stage. This stage starts with the reading the four successive coefficients from the Main Memory. The transform coefficients are stored in the main memory according to the Morton scanning order from lower addresses to higher addresses. In the preprocessing unit, these coefficients are read in the reverse of the Morton scanning order beginning from

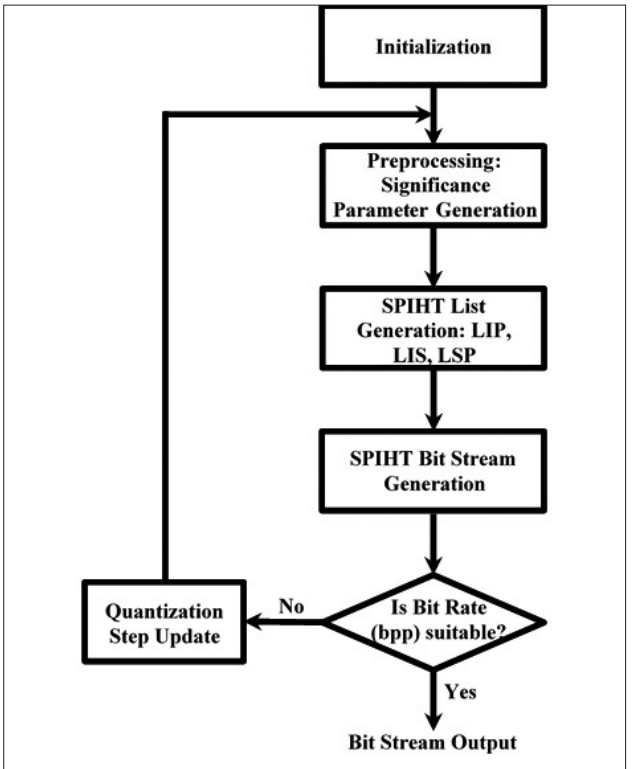

Figure 3. SPIHT encoder flow diagram

higher addresses. In this manner, data gathering starts from the lower right corner of the image, so that it is avoided that examination of the same pixels by visiting same positions.

Performing reading operation of the coefficients in this way provides considerable advantages in the significance tests of all descendants ($D(i,j)$) and all descendants excluding offspring ($L(i,j)$). If a coefficient is significant then all descendants of the parent and also all descendants of grandparent excluding offspring of that coefficient is significant. Since the coefficients in the SPIHT algorithm are arranged according to the quadtree structure, the same situation is also valid for the parent and grandparent of each four coefficients. By this means, reading of four coefficients from the main memory at the same time for the preprocessing stage provides efficiency in execution time. Moreover, four coefficients can be tested in parallel. Coefficient test unit fulfils the determination operation of five flags of a coefficient. These flag bits are defined as the significance of the coefficient, the sign bit if the coefficient is significant, significance of all descendants of the coefficient, the significance of all descendants excluding offspring of the coefficient, and the refinement bit. The flags of four coefficients are stored in the significance flag memory by aligning in accordance with the data structure given in Figure 5.

The flag data of concerned coefficient is read from the significance flags memory to find the significance flag value of all descendants or all descendants excluding offspring for a pixel. This data contains flags which possibly may have been written earlier belonging to offspring while examining offspring of that pixel.

Therefore, that data is necessary to calculate the significance flag of all descendants or all descendants excluding offspring for a coefficient. If the significance flag of a coefficient is 1, this situation affects the significance flags of all descendants of parent of the coefficient and all descendants excluding offspring of grandparent of the coefficient. Hence, the flag data stored in the significance flags memory of the parent pixels and grandparent pixels should be updated. The related fields of the data structure is updated by taking the flag data of parent pixel from memory, and data is stored to significance flags memory again. The same operation process is repeated for the grandparent pixel.

Another important unit at this stage is significance flags memory. The memory structure shown in Figure 6 also acts as pipeline register between pipeline stages since enhanced SPIHT encoder is designed to conform to pipelined datapath architecture. This designed memory structure is consisting of two individual memory unit, essentially. The use of data calculated in previous stage is made possible in the second stage while significance flag data is created in the first stage.

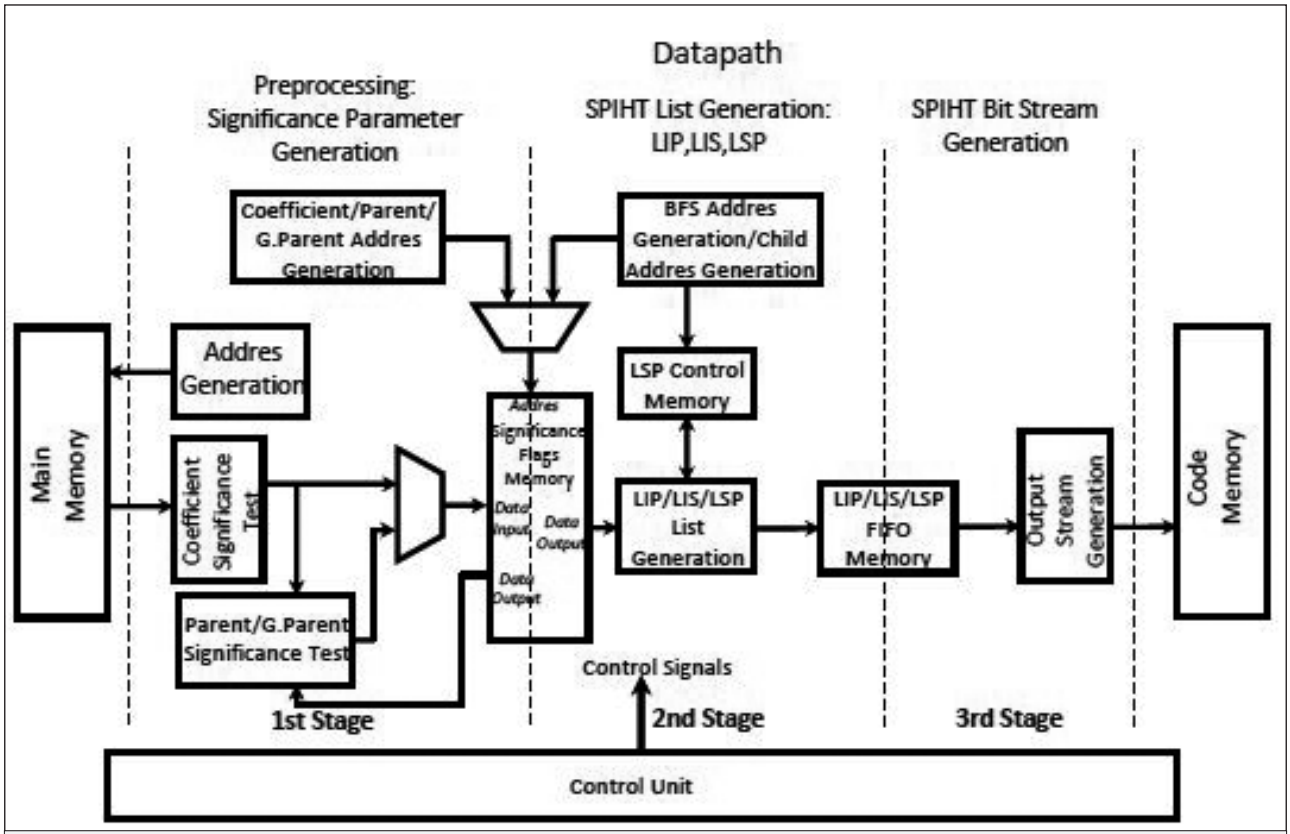

Figure 4. SPIHT encoder architecture

The second stage of the proposed architecture is generation of SPIHT Lists. Lists are constructed using flags generated in the first stage. Because of the quadtree structure and the fact that the flags are structured in quadtree form at first stage, the coefficients are also processed as quadtree form at this stage.

Initially, the flags of the first four coefficients are read. The second, third and fourth coefficients of the first four coefficients are the roots of quadtrees. Each quadtree is processed in parallel.

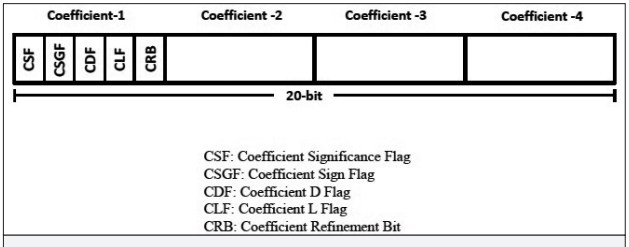

Figure 5. The data structure of coefficient significance flags

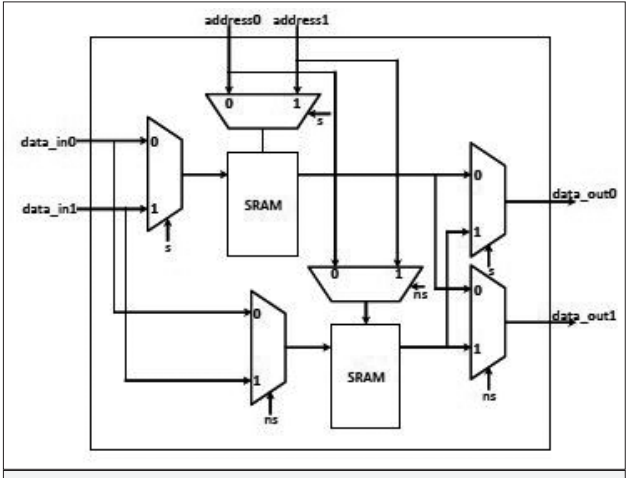

Figure 6. Design of significance flags memory (ns → ~s)

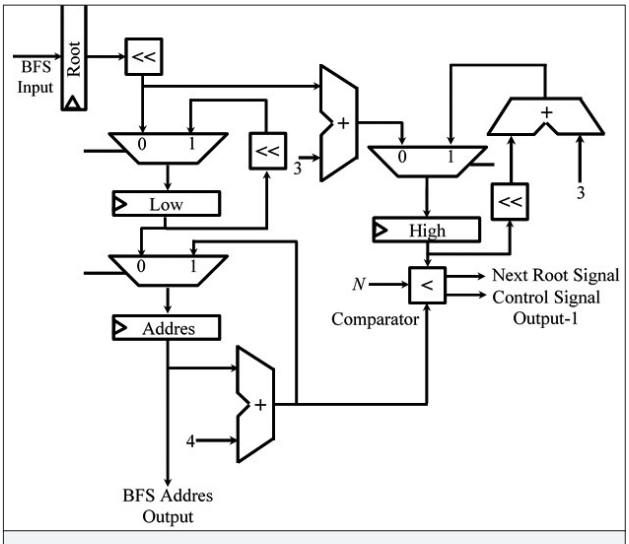

Figure 7. Design of BFS address generator

LIP, LIS and LSP are generated in FIFO (first in first out) memory structure. The size of lists is determined depending on image size and are not altered later. Some fields may stay empty in lists because all coefficients may not enter into these lists in an encoding stage. A 1-bit data, v-bit, is used to discriminate the empty and used fields. If this bit is 1, then this means that this field is used, and next bits will be coded in the output stream. There is a two-bit data field in LIP for each coefficient which enters this list except for the v-bit. These fields are the coefficient significance flag (CSF) and coefficient sign flag (CSGF). Two fields are defined in the LIS except for the v-bit. These fields are CDF and CLF bits. The CRB bit is existed in LSP except for the v-bit.

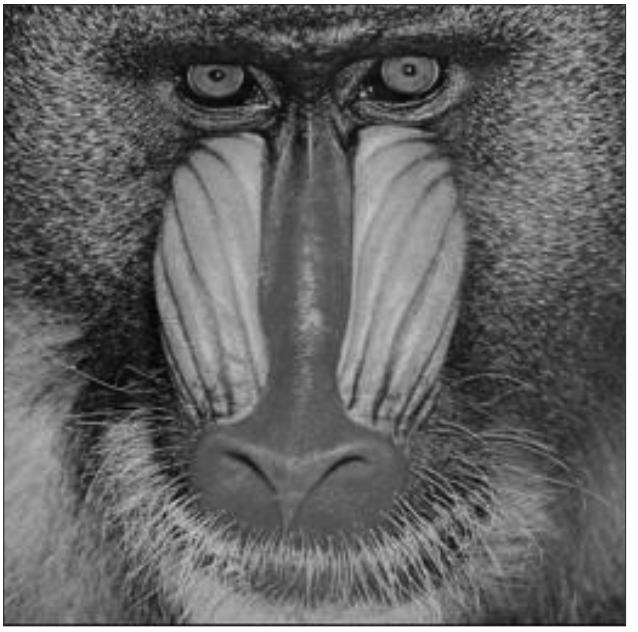

Figure 8. Test images

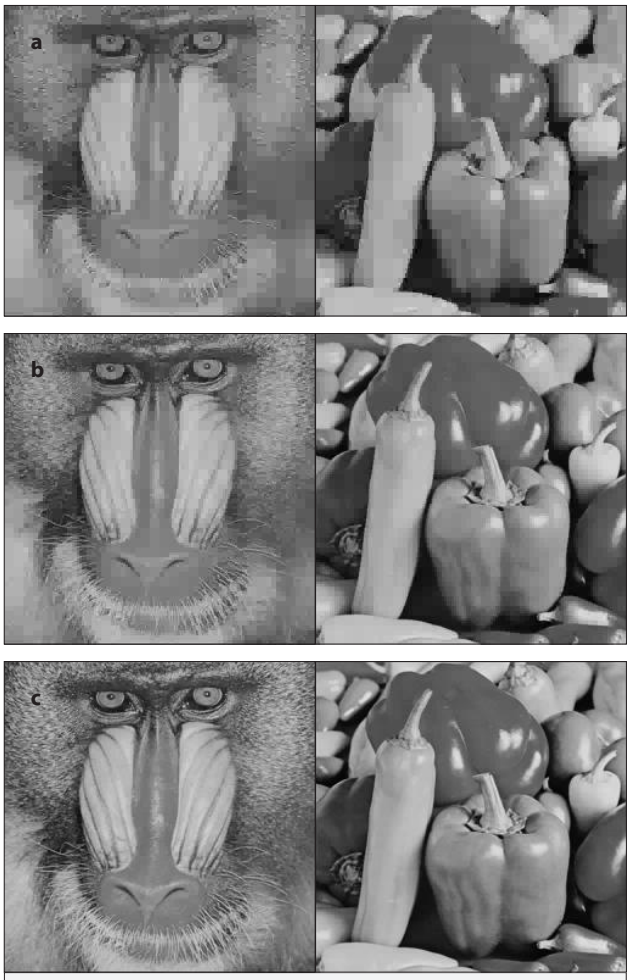

Figure 9. a-c. Reconstructed images for different bit rates. Reconstructed images for 0.035 bpp (a). Reconstructed images for 0.12 bpp (b). Reconstructed images for 0.44 bpp (c)

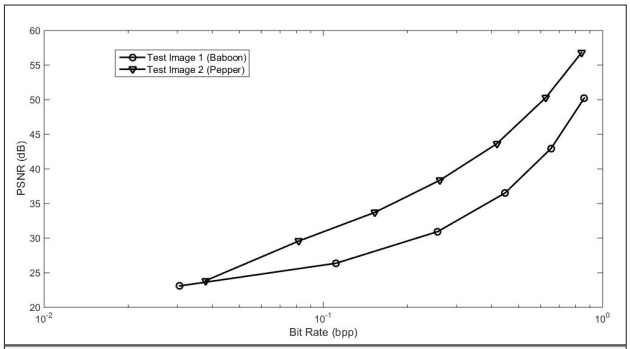

Figure 10. The variation of PSNR values with bit rate of the reconstructed images

Besides other list memories, another memory (LSP check) else is used to indicate which coefficient is inserted into LSP. The size of this memory is *NxN* bits. In LSP check memory, corresponding Morton order number bit of this coefficient is set as 1 when a coefficient enters into LSP. LIP and LIS memories are reseted at beginning of each quantization step. LSP and LSP check memories are used during the encoding without cleared. In other words, the entries in a quantization step are added following the entries of previous step. The memory structures of these lists are designed in complying with the memory structure used in the first stage and given in Figure 6.

Since the coefficients are processed using BFS in this stage the BFS address generator block given in Figure 7 is crucial.

Three registers are used in the design. Address of the coefficient at the root of the tree which will be scanned initially is recorded to the first register. The addresses of the coefficients on the left and right sides of these levels are recorded in the Low and High registers as the sublevels in the tree are examined by. The address scan starts from the Low register and continues to until the reach of the value in the High register. When the High register value is reached, the register values are updated according to the new level by descending one level down in the tree.

The third stage is the output stream stage. In the second step, list data stored in FIFO memory is transmitted to the output stream. Firstly, LIS FIFO memory, secondly LIP FIFO memory, and finally LSP FIFO memory contents are transmitted to the output stream.

Results

The designed SPIHT datapath has been tested using different images with 256×256 pixels. The wavelet coefficients of images at this size can be expressed by a 16-bit sign-magnitude representation.

First of all, the memory analysis necessary for encoding the images used in the tests has been performed. Since the coefficients and the output stream are stored in main memory, the memory space used for this data is not included in the analysis. Two types of memory have been used in the SPIHT datapath: The registers used for the intermediate operations, and SRAM between pipeline stages memories (significance flags memory and list memories). The analysis is performed for SRAM memory elements, since the memory space used for the registers is negligible as compared to the other memory space.

In significance flags memory, a 20-bit field is used for each of the four coefficients. Therefore, the total memory space used;

$$Significance\ Flags\ Memory\ Space = (2N^2/4)\times 20bit \quad (2)$$

If *N*=256 then this value is 80 Kbyte. The contents of LIP and LIS memories change at every calculation step. The size of the list memories has been determined by taking into account the maximum list size obtained in the tests performed and fixed for each calculation step. The number of coefficients entering the LIS increases as the threshold value decreases. The number of coefficients have the maximum value when the threshold value is equal to 1, and all the coefficients enter into this list except for the first one.

$$LIS\ Memory = 2(N^2 - 1) \times 3bit \quad (3)$$

If *N*=256 then this value is 48 Kbyte. The number of coefficients entering LIP memory has been determined by conducting different trials. As a result of the different trials, it has been concluded that the maximum size of LIP changes from 45% to 65% of number of coefficients. The size of the memory has been determined as given follows ensuring that not to overflow the memory.

$$LIP\ Memory = 2(0.75N^2) \times 3bit \quad (4)$$

If *N*=256 then this value is 36 Kbyte. LSP memory and LSP check memory contains all the coefficients, so the total LSP memory is given as;

$$LSP\ Memory = N^2 \times 3bit \quad (5)$$

If *N*=256 then this value is 24 Kbyte. Total memory space is 188 Kbyte if *N*=256.

The images used in the trials are given in Figure 8. The simulation of the design has been performed by storing the wavelet transform coefficients of these images to main memory. The output bitstreams for the different bit rates are also has been stored in the main memory. The images have been recovered by decoding the output bit streams in the decoder. The recovered images are shown in Figure 9. The PSNR criterion has been used to measure the similarity of the reconstructed images with the original image. The variation of PSNR values with bit rate of the reconstructed images is given in Figure 10.

Conclusion

In this study, the SPIHT image compression algorithm has been improved providing that the more compatible with hardware design process. The pipelined datapath design of the proposed enhanced SPIHT algorithm has been also performed. This datapath is comprised of three stages as preprocessing, list generation and output stream. As distinct from present studies, taking the coefficients in reverse Morton order avoids the unnecessary repeated operations for significance tests of all descendants and all descendants excluding offspring, in preprocessing stage. In list generation stage, the BFS has been considered, and a BFS address generator has been designed. The designed datapath has been tested by using different images. According to these tests, memory analysis of the datapath has been fulfilled and the obtained compression results has been given. The proposed method can be used for fast and less complex calculation required image compression systems in which low power and low capacity devices.

Peer-review: Externally peer-reviewed.

Conflict of Interest: The authors have no conflicts of interest to declare.

Financial Disclosure: The authors declared that this study has received no financial support.

References

1. J. M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", IEEE Trans Sig Proc, vol. 41, no. 12, pp. 3445-3462, 1993. [CrossRef]
2. A. Said, A.P. William, "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", IEEE Trans Circ Sys V Tech, vol.6, no.3, pp.243-250, 1996. [CrossRef]
3. D. Taubman, "High Performance Scalable Image Compression with EBCOT", IEEE Trans Im Proc, vol.9, no.7, pp.1158-1170, 2000. [CrossRef]
4. M. Akter, M. B. Reaz, F. Mohd-Yasin, F. Choong, "A Modified-Set Partitioning in Hierarchical Trees Algorithm for Real-Time Image Compression", Jour Com Tech Elec, vol. 53, no.6, pp. 642-650, 2008. [CrossRef]
5. Z. E. Hanaa, A. E. Mostafa, A. A. Hesham, "A Modified Listless Strip Based SPIHT for Wireless Multimedia Sensor Networks", Elsevier Comp Elec Eng, vol. 56, pp. 519-532, 2016. [CrossRef]
6. N. R. Rema, A. O. Binu, P. Mythili, "Image Compression Using SPIHT with Modified Spatial Orientation Trees", Elsevier Proce Comp Sci, vol. 46, pp.1732-1738, 2015. [CrossRef]
7. Y. Sun, H. Zhang, G. Hu, "Real-Time Implementation of a New Low-Memory SPIHT Image Coding Algorithm Using DSP Chip", IEEE Trans Im Proc, vol. 11, no. 9, pp. 1112-1116, 2002. [CrossRef]
8. L. W. Chew, L. M. Ang, K. P. Seng, "New Virtual SPIHT Tree Structures for Very Low Memory Strip-Based Image Compression", IEEE Sig Proc Lett, vol. 15, pp. 389-392, 2008. [CrossRef]
9. Y. Jin, H. J. Lee, "A Block-Based Pass-Parallel SPIHT Algorithm", IEEE Trans Circ Syst V Tech, vol. 22, no. 7, pp.1064-1075, 2012. [CrossRef]
10. H. J. O. Dominguez, O. O. V. Villegas, V. G. C. Sanchez, "Modified Set Partitioning in Hierarchical Trees Algorithm Based on Hierarchical Subbands", Jour Elec Im, vol. 24, no. 3, pp.1-12, 2015.
11. S. Cekli, A. Akman, "An Efficient SPIHT Algorithm and System Architecture for Image Compression", Sig Proc Com App Conf (SIU), 15-18 May, Turkey, 2017. [CrossRef]
12. J. H. Hsieh, R. C. Lee, K. C. Hung, M. J. Shih, "Rapid and Coding-Efficient SPIHT Algorithm for Wavelet-Based ECG Data Compression", Elsevier Int, VLSI Jou, vol. 60, pp. 248-256, 2018. [CrossRef]
13. T. W. Fry, S. A. Hauck," SPIHT Image Compression on FPGA", IEEE Trans Circ Syst V Tech, vol.15, no. 9, pp.1138-1147, 2005. [CrossRef]
14. S. Kim, D. Lee, J. S. Kim, H. J. Lee, "A High-Throughput Hardware Design of a One-Dimensional SPIHT Algorithm", IEEE Trans Multimedia, vol. 18, no. 3, pp. 392-404, 2016. [CrossRef]
15. T. Hadjem, M. S. Azzaz, C. Tanougast, S. Sadoudi, "A New Image Crypto-Compression System SPIHT-PSCS", Int Conf Cont, Dec Inf Tech (CoDIT), 3-5 November, France, 2014. [CrossRef]
16. M. Zhang, X. Tong, "Joint Image Encryption and Compression Scheme Based on IWT and SPIHT", Elsevier Opt Lasers Eng, vol. 90, pp. 254-274, 2017. [CrossRef]
17. Z. Lu, D. Y. Kim, W. A. Pearlman, "Wavelet Compression of ECG Signals by the Set Partitioning in Hierarchical Trees Algorithm", IEEE Trans Bio Eng, vol. 47, no. 7, pp. 849-856, 2000. [CrossRef]

Serap Çekli was born in Germany in 1978. She received her BSc. degree in Electronics Engineering from Istanbul University in 2000, MSc. degree in Electronics and Communication Eng. from Istanbul Technical University in 2003. She has received PhD. degree from Istanbul University, Elect. Electronics Eng. Dept. in 2009. She worked for Istanbul University Engineering Faculty as a research assistant between2001-2009. She has been working as a asistant professor at Maltepe University, Computer Engineering Department since 2009. Her research interests are digital systems, digital design, computer organization and architecture.

Ali Akman received her BSc., MSc. and PhD. degrees in Electronics Engineering from Uludağ University, Bursa, Turkey in 1993, 1996 and 2004, respectively. He has been working as a asistant professor at Maltepe University, Computer Engineering Department since 2007. Pirior to joining Maltepe University, he worked for Uludağ University, Department of Electronics Engineering as a research assistant and lecturer. His research interests are digital systems, computer architecture, embedded systems, networked embedded systems and wireless sensor networks.