Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 53(67), Fascicola 2, 2008

Efficient Architecture for SPIHT Algorithm for Compression of Images

Anil V. Nandi[1], R.M.Banakar[2]

Abstract – Our work involves synthesis and FPGA implementation of high speed and high throughput Superscalar 'Set Partitioning in Hierarchical Trees' (SPIHT) algorithm for compression of natural images. Because of its inherent redundancy removal property among wavelet coefficients SPIHT is well suited for compression of both gray and color images. But the basic SPIHT algorithm uses dynamic data structures which hinders hardware realization. In our implementation we have used modified SPIHT. Modifications are in two ways, one by using static (fixed) mappings which represent significant information and the other by inter-changing the sorting and refinement passes. The implementation involves pipelining and parallelism operations in SPIHT blocks and peripheral circuitry. The address generation unit is designed in such a way that the coefficients of different sub bands are accessed from memory efficiently and helps in achieving higher speed and throughput. A hardware realization is done in a Altera Cyclone II FPGA development board. Significant clock speed and throughput are obtained for a test image of size 128 x128 pixels.

Keywords: Compression, SPIHT, Pipeline, Parallel

I. INTRODUCTION

DWT of an image leads to its representation indifferent scales in the form of Spatial Orientation Trees (SOT). The coefficients in SOT of an image DWT are encoded using embedded zero-tree coding which is proved to be the best image compression technique [1][2]. After the transform, the lowest frequency coefficients concentrate most of the energy of the transformed image. The high frequency coefficients of different scales and orientations indicate the strong self similarity among themselves. These properties are exploited in SPIHT [1]. After EBCOT in JPEG2000, SPIHT a sophisticated coding technique belongs to next generation of encoders for wavelet transformed images. The basic SPIHT uses dynamic data structures for exploiting self similarities mentioned above. These dynamic data structures impose practical limitation on hardware implementation of SPIHT, unlike software implementation where dynamic data structures can be implemented conveniently using linked lists [3]. Hence we have modified the basic SPIHT algorithm to overcome dynamic allocation problem. The FPGA implementation of modified version has resulted in significant optimization in memory requirements and speed.

The paper is organized as follows. In Section II and III, descriptions of the basic SPIHT algorithm and the modified SPIHT algorithm are described respectively. Section IV presents hardware architecture and implementation of modified SPIHT. Results and conclusions are discussed in section V and VI respectively.

II. SPIHT ALGORITHM

The SPIHT algorithm uses a partitioning of the spatial orientation trees in a manner that tends to keep insignificant coefficients together in larger subsets. The partitioning decisions are binary decisions that are transmitted to the decoder, providing a significance map encoding. The thresholds used for checking significance are powers of two, so in essence the SPIHT algorithm sends the binary representation of the integer value of the wavelet coefficients. The significance map encoding or set partitioning and ordering step is followed by a refinement step in which the representations of the significant coefficients are refined. The SPIHT algorithm can be applied to both gray-scale and colored images. SPIHT displays exceptional characteristics over several properties like good image quality, fast coding and decoding, a fully progressive bit stream, application in lossless compression, error protection and ability to code for exact bit rate. The SPIHT process represents a very effective form of coding. A straightforward consequence of the compression simplicity is the greater coding/decoding

[1]Assistant Professor,[2] Professor and Member-IEEE, Electronics and Communication Engineering Department, B.V.Bhoomaradi College of Engg. & Tech., Vidyanagar, Hubli-580 031, Karnataka, INDIA. e-mail anilnandy@bvb.edu

BUPT

speed. The SPIHT algorithm is nearly symmetric, i.e. the time to encode is nearly equal to the time to decode. SPIHT codes the individual bits of the image wavelet transform coefficients following a bit plane sequence. Thus, it is capable of recovering the image perfectly by coding all bits of the transform. In practice it is frequently possible to recover the image perfectly using rounding after recovery, but this is not the most efficient approach. SPIHT due to its embedded coding property is much easier to design efficient error-resilient schemes. This happens because with embedded coding the information is sorted according to its importance, and the requirement for powerful error correction codes decreases from the beginning to the end of the compressed file. If an error is detected, but not corrected, the decoder can discard the data after point and still display the image obtained with the bits received before the error. Another reason is that SPIHT generates two types of data. The first is sorting information, which needs error protection. The second consists of uncompressed sign and refinement bits, which do not need special protection because they affect only one pixel.

III. MODIFIED SPIHT ALGORITHM

The SPIHT algorithm [1] computes the SOT wavelet coefficients progressively in dynamic order. The work by Singh et. al. [4] uses content addressable memories to keep track of the order in which the coefficients are scanned. Algorithm is implemented directly without modification not taking into account hardware optimization. If the bit streams end within the middle of the bit-plane, then the quality of the image will be better for such a scheme. Since every image has a unique order of coefficients determined by their values, the generation of bit-stream depends on 2x2 block of coefficients, their children and maximum value within that sub-tree. So, every block of coefficients can be operated independent of others and also in parallel to one another. However the order in which they are to be operated is not known in advance, due to uniqueness of every image. The basic SPIHT algorithm determines the ordering of coefficients in sequential manner. The computation traverses the coefficients of an image many times in each bit plane and inserts or deletes the coefficients in the lists. Such a scheme is not suitable for hardware implementation using parallelism and limits the throughput. Our scheme uses fixed order SPIHT [5], in which the order in which the block of coefficients are transmitted is fixed in advance. The blocks of coefficients are inserted in the predetermined order. The order is based upon Morton Scan ordering [6]. This eliminates the overhead of computing the order of coefficients in each bit-plane. The parallel units of SPIHT can be built with this technique so that throughput can be increased. Another modification is in the refinement pass. In basic SPIHT, the information regarding status for the elements of LSP, whether they have been added in the current iteration of sorting pass, need to be maintained. In the worst case if all the coefficients become significant in the same iteration, we need large memory requirement to store this information. If we exchange the sorting pass and refinement pass we need not store this information. The data stream is still decodable and does not increase in size. We need to consider the reordering of the transmitted bits during the iteration in the decoding process.

A. The modified SPIHT algorithm is as follows:

Initialization
n = log2 (max |coeff|)
LIP = All elements in H
LSP = Empty
LIS = D's of Roots
Refinement Pass
Process LSP
for each element (i, j) in LSP – except those just added
above.
Output the nth most significant bit of coefficient
End loop over LSP
Significance Map Encoding ("Sorting Pass")
Process LIP
for each coeff (i, j) in LIP
Output Sn (i, j)
If Sn (i, j)=1
Output sign of coeff (i, j): 0/1 = -/+
Move (i, j) to the LSP
End if
End loop over LIP
Process LIS
for each set (i, j) in LIS
if type D
Send Sn(D (i, j))
If Sn (D (i, j))=1
for each (k, l),O (i, j)
output Sn (k, l)
if Sn (k, l)=1, then
add (k,l) to the LSP and output sign of coeff:0/1+/-
if Sn (k,l)=0, then add (k,l) to the end of the LIP
end for
end if
else (type L)
Send Sn (L(i,j))
If Sn (L(i,j))=1
add each (k,l),O (i,j) to the end of LIS as an entry of type D
remove (i,j) from the LIS
end if on type

BUPT

End loop over LIS
Update
Decrement n by 1
Go to Significance Map Encoding Step

B. Address generation unit:

DWT decomposed coefficients arrangement used in the SPIHT algorithm is shown in Fig. 1. The arrows indicate the descendants of a particular wavelet coefficient. Searching for descendants of a specific coefficient takes lots of operations. To access the descendents efficiently, it is necessary to store 2-D data in a 1-D array using a dedicated hardware. There are various suggestions for SPIHT modifications based on parallelism and memory issues [5, 7].

We can use 8 bit symbols to represent 2-D addresses. We have used our novel method to calculate the addresses of the direct offspring's of a coefficient. The search for the descendants of a specific coefficient becomes very easy in this scheme. In this scheme, the direct offspring's are stored in consecutive memory addresses, which make the memory-read operation more efficient and reduce the switching frequency of the address bus. The address generation circuit is simply a shifter and an increment-by-one operation. Furthermore, the above relation does not change when the image dimension is changed. This modification is done to simplify the hardware design without much loss in coding efficiency. All the definitions in the original SPIHT algorithm are identical except the 2-D addresses being replaced by 1-D addresses.

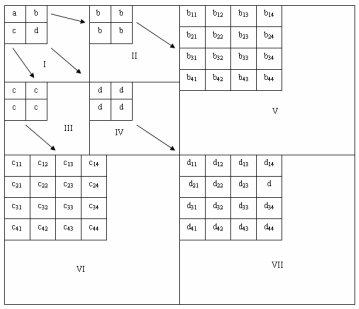

Fig 1 DWT decomposed coefficients

The hardware implementation of modified SPIHT is necessary due to requirements in higher speed, throughput, optimized area and low power. These are very difficult to achieve in software-based approach implemented on a general processor. The direct implementation of SPIHT software algorithm is briefly explained in [4]. It provides brief overview of the architecture and does not discuss the qualitative results.

After Bi-Orthogonal (9,7) DWT decomposition, the coefficients are obtained. We have assumed coefficients to be in the text file and the address generator generates the addresses so as to pick up the coefficients and place them in the coefficients memory. The coefficients are stored in the sequential manner in the main memory. The architecture of memory address generator block is shown in Fig 2. The coefficients are read into the lists of SPIHT encoder and are compared with the threshold value and the corresponding code is generated from code generator.

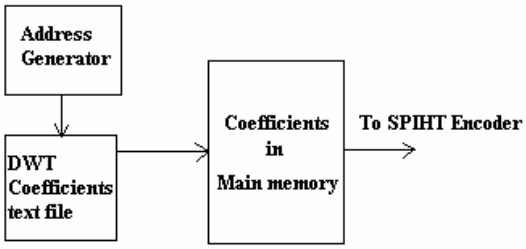

Fig 2 Memory Address Generator block

IV. ARCHITECTURE

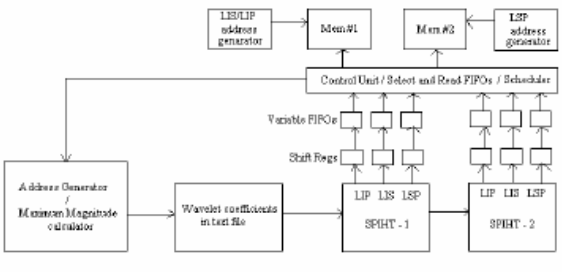

Fig 3 Fixed Order Parallel SPIHT block diagram

Fig 3 shows Fixed order Parallel architecture.

A. Maximum magnitude calculator / Address generator block:

The hierarchical tree structure obtained after wavelet decomposition has a particular pattern for accessing the coefficients. The relation between the row and the column values of a parent co-ordinate and its four descendants are as follows. Suppose the parent be 'a' and its row and column address be X[0] and Y[0] respectively. Then the row and column addresses for its descendants a1, a2, a3, a4 are as given below.

a1: X[1]= 2X[0], Y[1]= 2Y[0]
a2: X[2]= X[1], Y[2]= Y[1]+1
a3: X[3]= X[1]+1, Y[3]= Y[2]-1
a4: X[4]= X[3], Y[4]= Y[3]+1

BUPT

Using the same pattern, the tree grows on for each of the nodes and it stops at the last element of the matrix. The address generator has been designed to implement the above mentioned pattern in hardware. Thus the random access of coefficients is converted into sequential access. Maximum magnitude coefficient is computed by searching all the coefficients of the image matrix using sequential addressing for the wavelet coefficients. This process of finding the maximum value of coefficient slows down the system performance. By using Depth First Search order algorithm [8] performance can be improved. Maximum magnitude calculation phase calculates the maximum value of coefficients and rearranges the following information for SPIHT block. a) Maximum magnitude of four child trees b) Current maximum magnitude c) Threshold and sign data of each of the 16 child coefficients 4) Reorder the coefficients into sequential order. The address generator also generates the addresses fixed order addresses and picks the data (wavelet coefficients) from text files. We have used text file storage for wavelet coefficients. In real time operation, they are fed from the wavelet transform block. The coefficients are fed to the SPIHT blocks (SPIHT-1 and SPIHT-2) in the predefined order.

B. SPIHT block:

We have implemented a SPIHT coding block consisting of three parallel units. These three units code the data arriving from magnitude calculator/Address generator block. Coefficient blocks are read from higher level to lower level. The fixed point numerical representation is used for each wavelet level. When the three blocks receive data in common format, parallel version of SPIHT operates to generate code corresponding to information in each block that contributes to each bit-plane. Each SPIHT block generates bit stream which are added and grouped before sending to the variable FIFOs for each bit-plane. The data received in FIFOs vary in size depending on the coefficients values in each bit-plane. Maximum size of FIFO is kept at 20 bits after verification by software program [3] for a class of 128x128 pixel images. Variable FIFOs arrange the block of data into regular size of 16 bit words for memory access. Scheduler is used to take care by stalling the algorithm operation if one of the FIFOs becomes full. The scheduler determines the filled FIFO and writes the filled data into memory #1 and memory #2.

C. Platform for Implementation:

Hardware is modeled using VHDL under Quartus II tools [9] with Modelsim simulator. Cyclone II FPGA reconfigurable system is used for implementation.

V. RESULTS AND DISCUSSION

A. Memory requirement of SPIHT:

The size of required working memory at any instant of time depends on the number of entries in the lists. The memory requirements at the end of each pass (bit-plane) and maximum (worst case) memory requirement are considered for memory requirement. The three lists LIP, LIS and LSP are used in SPIHT. Each entry in the lists uses co-ordinates of wavelet coefficients. The memory requirements are calculated as follows:
The total memory required for SPIHT be MSPIHT.
MSPIHT= c. NLIP + (c-1) NLIS + c NLSP
where NLIP, NLIS and NLSP are number of elements in the lists LIP, LIS and LSP respectively.
Worst case is when
NLIP + NLSP = M.N. (for M x N image) and
NLIS = M.N / 4
Considering the worst case when, High frequency sub-bands will never enter into LIS.
MSPIHT (max) = (5C-1) (M.N) / 4.
For 128 x128 image and c=9 bits, MSPIHT (max) will be 704 bits.

B. Clock rates and Throughput - SPIHT:

The encoding clock cycles for SPIHT with full pipeline and three parallel units for 128x128 test image (Lena, Gray) is 689 cycles. It does not include the cycles for Maximum Magnitude calculation and address generation block. The maximum clock rate available on the board is 25 MHz. The average number of bits generated at the output of Parallel SPIHT block is 2.9 bits per clock cycle as compared to 2.1 bits per clock cycle for two parallel units without pipeline. The Table 1 shows the results

Phase – SPIHT	**Cycles/128x128 image**	**Clock rate**	**Throughput**
Without pipeline			
Two parallel units	1030	10 MHz	30 MPixels/Second
Three parallel units	790	10 MHz	37 MPixels/Second
With pipeline			
Two parallel units	810	20 MHz	36 MPixels/Second
Three parallel units	689	20 MHz	43 MPixels/Second

Table 1. Results for SPIHT block

BUPT

C. Clock rates and Throughput–Address generation unit:

The test image used is of the order 128x128 pixels. The FPGA resources, maximum clock frequency and throughput corresponding to 1-D and 2-D memory schemes are shown in the Table 2. The clock cycles do not include the cycles for Maximum magnitude calculation.

Memory	Area (FPGA) FFs	LUTs	RAM	Maximum clock frequency	Throughput
2-D memory scheme	12%	14%	10%	27MHz (1870 clock cycles)	18 MPixels /second
1-D memory scheme	12%	14%	8%	45MHz (1579 clock cycles)	29 MPixels /second

Table 2. Results for 2-D (Original SPIHT) and 1-D (modified SPIHT) memory accesses

D. PSNR measurement:

The output of pipelined and parallel SPIHT coding block consists of similar bit streams as that of basic SPIHT (without fixed order) but in different order. PSNR at the end of each bit-plane will be closely matching to that of regular SPIHT as shown in the Fig 4. The slight difference is observed due to short length of each bit stream in fixed order scheme. At rates below about 0.55 bpp, the difference falls within 0.18 db. At lower bit rates the PSNR performances are equal. The difference is found to be increasing at higher rates. The proposed modification is applicable for applications requiring higher compression ratios. Maximum loss is found to be 0.18 dB over the shown bit rates.

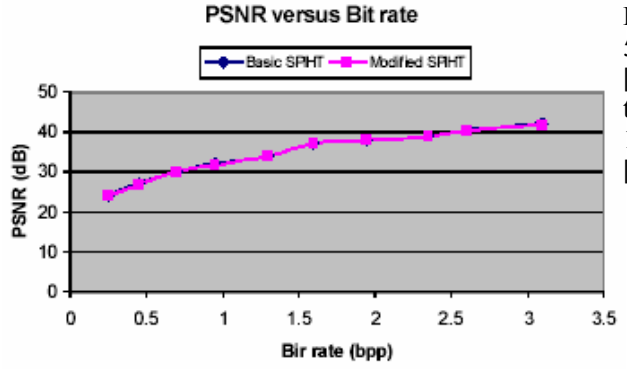

Fig 4 PSNR versus Bit rate

VI. CONCLUSIONS

We have presented a hardware efficient image codec based on modified SPIHT. Efficient coefficient addressing method, fixed-size lists and exchange of passes are used in the implementation. SPIHT provides easy rate control and no need of look-up table. The throughput with full pipeline and three parallel SPIHT units is found to be increased by 38% as compared to two parallel SPIHT units without pipeline The speed (clock rate) and throughput are increased to a greater extent with 1-D addressing scheme as compared to that of 2-D addressing scheme. The speed is increased by 66.67% and throughput is increased by 61.11%.

REFERENCES

[1] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees in Trans. Signal Processing, volume 5, no.9, pages 1303–1310. IEEE, 1996.
[2] I. Daubechies. Ten lectures of wavelets. SIAM, Philadelphia PA, 1992.
[3] Priyanka G,Shalini Neeli,Trupti R,Kavya P, Anil V. Nandi, R.M.Banakar, ImaegeCodec using Wavelet/Set Partitioning In Hierarchical Tree approach, National Conference on Signal Processing and Communications-2006, (NCSPC-2006), at JNNCE, Shimoga.
[4] J Singh, A. Antoniou, D. J. Shpak, "Hardware Implementation of a Wavelet based Image Compression Coder," IEEE Symposium on Advances in Digital Filteringand Signal Processing, pp 169 – 173, 1998.
[5] Thomas W. Fry, Scott Hauck, "SPIHT image compression on FPGAs", IEEE Transactions on circuits and systems for video technology, Vol 15, No.9, September 2005 pg 1138-1147.
[6] V. R. Algazi, R. R. Estes. "Analysis based coding of image transform and sub band coefficients," Applications of SPIE Proceedings, pages 11-21, 1995.
[7] Maurizio Martina, Andrea Molino, Andrea Terreno and Fabrizio Vacca, "Implementation of SPIHT coprocessor: Memory issues and Hardware implications", Dipartimento di Elettronica, Politecnico do Torino – ITALY, 2003 IEEE, pg. II-587 to II-590.
[8] T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, MIT Press, Cambridge, Massachusetts, 1997.
[9] Quartus II user manual.

BUPT