High Performance Hardware Architectures for One Bit Transform Based Motion Estimation

Abdulkadir Akin, Yigit Dogan, and Ilker Hamzaoglu, *Member,* IEEE

Abstract — *Motion Estimation (ME) is the most computationally intensive part of video compression and video enhancement systems. One bit transform (1BT) based ME algorithms have low computational complexity. Therefore, in this paper, we propose high performance systolic hardware architectures for 1BT based fixed block size (FBS) and variable block size (VBS) ME. The proposed ME hardware architectures perform full search ME for 4 Macroblocks in parallel and they are faster than the 1BT based ME hardware reported in the literature. They use less on-chip memory than the previous 1BT based ME hardware by using a novel data reuse scheme and memory organization. The proposed VBS ME hardware is the first 1BT based ME hardware implementing VBS ME. The proposed hardware architectures are implemented in Verilog HDL. The FBS ME and VBS ME hardware consume %34 and %49 of the slices in a Xilinx XC2VP30-7 FPGA, respectively. They can work at 113 MHz in the same FPGA and are capable of processing 49 1920x1080 full High Definition frames per second. Therefore, they can be used in consumer electronics products that require real-time video processing or compression.*[1]

Index Terms — Motion Estimation, Variable Block Size Motion Estimation, One-Bit Transform, Hardware Implementation, FPGA.

I. INTRODUCTION

Motion Estimation (ME) is the most computationally intensive part of video compression and video enhancement systems. ME is used to reduce the bit-rate in video compression systems by exploiting the temporal redundancy between successive frames, and it is used to enhance the quality of displayed images in video enhancement systems by extracting the true motion information. ME is used in video compression standards such as MPEG4 and H.264 [1], and in video enhancement algorithms such as frame rate conversion [2, 3].

Block Matching (BM) is the most preferred method for ME. BM partitions current frame into non-overlapping NxN rectangular blocks and tries to find a block from a reference frame in a given search range that best matches the current block. Sum of Absolute Differences (SAD) is the most preferred block matching criterion. The SAD value of a search location defined by the motion vector $d(d_x,d_y)$ is calculated as in (1), where $c(x,y)$ and $r(x,y)$ represent current and reference frames, respectively. The coordinates (i,j) denote the offset locations of current and reference blocks of size NxN.

$$SAD(\vec{d}) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} \left| c(x+i,y+j) - r(x+i+d_x,y+j+d_y) \right| \quad (1)$$

Among the BM algorithms, Full Search (FS) algorithm achieves the best performance since it searches all search locations in a given search range. However, the computational complexity of FS algorithm for fixed block size (FBS) ME is very high. In order to improve the performance of ME, variable block size (VBS) ME is used in H.264 standard. The computational complexity of FS algorithm for VBS ME is even higher [4, 5].

Several fast search ME algorithms, such as New Three Step Search (NTSS) [6], Diamond Search (DS) [7], Hexagon-Based Search (HEXBS) [8], and Adaptive Dual Cross Search (ADCS) [9], are proposed to reduce the computational complexity of FS algorithm. These algorithms try to approach the PSNR of FS algorithm by computing the SAD values for fewer search locations in a given search range. Several hardware architectures for fast search ME algorithms are proposed in the literature [10, 11].

Another preferred method for reducing the computational complexity of FS algorithm is reducing pixel resolution from 8 bits to fewer bits. In [12], the one-bit transform (1BT) technique is proposed to reduce the computational complexity of the matching process in ME by transforming video frames into 1 bit/pixel representations and performing ME using these binary representations. Although an 8-bit SAD calculation requires a subtraction and absolute value operation, 1-bit matching only requires an exclusive-or (XOR) operation and is very suitable for hardware implementation.

In [12], video frames are filtered using a multi-bandpass filter and the filtered results are used as pixel-wise thresholds to construct the binary representations used for ME. In [13], a new multi-bandpass filter kernel is proposed for 1BT to facilitate a multiplication free transform for reduced transform complexity. An early termination scheme for binary ME is presented in [14]. In [15], two bit transform (2BT) is proposed to improve ME accuracy compared to 1BT by constructing two bit-planes for each video frame and performing ME using 2BT representations. In [16], constraint one-bit transform (C-1BT) is proposed and it is shown that C-1BT provides increased ME accuracy compared to 2BT at much lower complexity.

The first 1BT based ME hardware implementation in the literature is presented in [12]. In [12], a motion vector (MV)

[1] This research was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK) under the contract 107E179.

A. Akin, Y. Dogan and I. Hamzaoglu are with the Department of Electronics Engineering, Sabanci University, Istanbul, 34956, Turkey (e-mail: {abdulkadir, yigitdogan}@su.sabanciuniv.edu, hamzaoglu@sabanciuniv.edu).

Contributed Paper
Manuscript received April 15, 2009

based linear arrays hardware architecture is used for implementing 1BT based ME. In [17], a source pixel based linear arrays hardware architecture is used for implementing low bit depth ME algorithms proposed in [13] and [16]. In [18], a new sub-pixel accurate low bit depth ME algorithm and its hardware is presented.

Data reuse method is used for reducing the off-chip and on-chip memory bandwidth required by ME hardware [19, 20]. In [19], a redundancy access factor and four levels of data reuse are provided to measure the degree of redundancy and a FS hardware architecture with a low memory bandwidth is proposed. In [20], search window (SW) data flows both horizontally and vertically while the current block is stationary and processing element design ensures highly regular dataflow through processing elements avoiding long interconnect delays.

In this paper, we propose high performance systolic hardware architectures for 1BT based FBS ME and VBS ME. The proposed 1BT ME hardware architectures are based on the 8 bits/pixel FBS ME hardware architecture proposed in [11]. The major differences between them are the proposed ME hardware architectures calculate MVs of 4 Macroblocks (MBs) in parallel, use a novel data reuse scheme, and use less on-chip memory, processing element array and adder tree area because of 1BT.

The proposed ME hardware architectures are faster than the 1BT based ME hardware architectures reported in the literature and they are capable of processing 1920x1080 full High Definition (HD) videos in real-time. The 1BT based ME hardware proposed in [12, 17] cannot process 1920x1080 full HD videos in real-time. The Non-Match Counter architecture used in the proposed ME hardware is faster and has smaller area than the Non-Match Counter architecture used in the ME hardware proposed in [12, 17]. Although the proposed ME hardware store search windows of 4 MBs in on-chip memory, they use less on-chip memory and they load the on-chip memory from off-chip memory in less number of clock cycles than the ME hardware proposed in [12, 17].

The proposed FBS ME and VBS ME hardware architectures are implemented in Verilog HDL. The Verilog RTL codes are verified by simulation using Mentor Graphics Modelsim and are mapped to Xilinx *XC2VP30-7* FPGA using Xilinx ISE. FBS ME hardware consumes 4758 slices (%34 of all the slices) and 8 BlockRAMs (BRAMs). VBS ME hardware consumes 6782 slices (%49 of all the slices) and 8 BRAMs. Both FBS ME and VBS ME hardware can work at 113 MHz, and they are capable of processing 49 1920x1080 full HD frames per second. Therefore, they can be used in consumer electronics products that require real-time video processing or compression.

The rest of the paper is organized as follows. Section II explains 1BT based ME. The proposed systolic hardware architectures for 1BT based FBS ME and VBS ME are described in Section III and Section IV respectively. The implementation results are given in Section V. Section VI concludes the paper.

II. 1BT Based Motion Estimation

In [12], a multi-band pass filter that has 25 non-zero elements is used to obtain filtered images. The filtered images are compared to the original images to create the one-bit images. In this case, non-integer operations are required for the normalization stage of the filtering which has comparatively higher computational complexity. In [13], a novel diamond shape kernel (2) is proposed to decrease the computational complexity of the filtering stage of 1BT. This new kernel contains a power of 2 non-zero elements and thus the multiplication operation becomes simple logical shift. For filtering, sixteen additions and one four-bit shift operation is performed.

$$K = \frac{1}{16}\begin{bmatrix}
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0
\end{bmatrix} \tag{2}$$

$$B(i,j) = \begin{cases} 1, & if\ I(i,j) \geq I_F(i,j) \\ 0, & otherwise \end{cases} \tag{3}$$

$$NNMP(m,n) = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} B^t(i,j) \oplus B^{t-1}(i+m, j+n) \tag{4}$$

Therefore, this method is called multiplication free one-bit transform (MF1BT). In MF1BT, the standard bit-plane is obtained as in conventional 1BT as shown in (3). The number of non-matching points (NNMP) criterion proposed in [12] is then used to evaluate the match of two blocks as shown in (4). The symbol ⊕ denotes XOR operation. The search location which has the smallest *NNMP* value is selected as the MV of the current block.

The FBS ME and VBS ME hardware architectures proposed in this paper implement both the 1BT based ME proposed in [12] and MF1BT based ME proposed in [13].

III. Proposed Hardware Architecture For 1BT Based Fixed Block Size Motion Estimation

The block diagram of the proposed hardware for 1BT based FBS ME is shown in Fig. 1. The hardware has 8 BRAMs, Vertical Rotator, One Bit Selector, 4 Processing Element (PE) arrays, Control Unit, Comparator & MV Generator. The hardware finds MVs of 4 16x16 MBs in parallel using full search

ME algorithm based on minimum NNMP criterion in a search range of [-16, 16] pixels. Its latency is 6 clock cycles; one cycle for synchronous read from memory, one cycle for Vertical Rotator and One Bit Selector, one cycle for Non-Match Counter, two cycles for Adder Tree and one cycle for Comparator & MV Generator. The Control Unit generates the required address and control signals to compute the NNMP values of the search locations in the search windows of the 4 MBs.

Search windows of 4 16x16 MBs (MB0, MB1, MB2 and MB3) and their search locations for [0,0] MV are shown in Fig. 2. Total search window (SW_T) size for 4 MBs is 48x96 pixels. There are large intersections between the search windows of these MBs, e.g. 2/3 of the SWs of MB3 and MB2 are the same and 1/3 of the SWs of MB3 and MB1 are the same. Therefore, performing ME for these 4 MBs in parallel allows significant data reuse.

The search locations in a SW_T are searched line by line and the search locations in each line are searched from right to left. MB PE arrays start at the same time by searching their right most search locations in the first line of the SW_T and finish at the same time after searching their left most search locations in the line 32 of the SW_T. The first search location searched by MB3 includes the SW pixels 0 to 15 in the lines 0 to 15. The first search locations searched by the other three MBs include the SW pixels 16 to 31, 32 to 47, 48 to 63 respectively in the lines 0 to 15. After MB PE arrays finish searching their left most search locations in a line, they search the search locations in the next line of the SW_T starting from their right most search locations in that line.

Comparator & MV Generator compares the NNMP values computed by each PE array and determines the minimum NNMP value and the corresponding MV for each MB (MB0, MB1, MB2 and MB3).

A. Systolic PE Array and Data Reuse Scheme

There are 256 PEs in each PE array. The architecture of MB1 PE array is shown in Fig. 3. After a PE array computes same line. The SW pixels needed for computing the NNMP values for first search locations of 4 MBs in a line are loaded from BRAMs into PE arrays. PE arrays, then, reuse SW pixels for computing the NNMP values for the neighboring search locations in a line. The same current MB pixel is used by a PE while computing NNMP values for $(16+16+1)^2 = 1089$ search locations.

Each PE is connected to its neighboring PE in order to shift the SW pixel to right by one. Therefore, each PE array needs 16 new SW pixels for computing the NNMP value for the next search location. The 16 new SW pixels needed by MB3 PE array for computing the NNMP value for the second search location in the first line are pixel 16 in the lines 0 to 15. MB3 PE array gets the 16 new SW pixels it needs from MB2 PE array. Similarly, MB2 PE array gets the 16 new SW pixels it needs from MB1 PE array and MB1 PE array gets the 16 new SW pixels it needs from MB0 PE array. MB0 PE array gets the 16 new SW pixels from One Bit Selector.

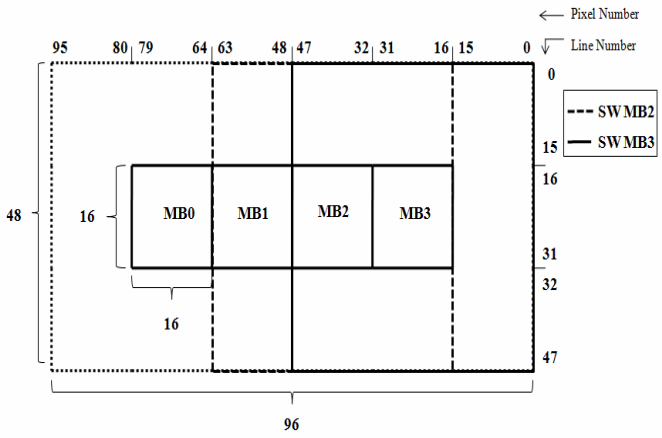


Fig. 2. Search Windows of 4 Macroblocks

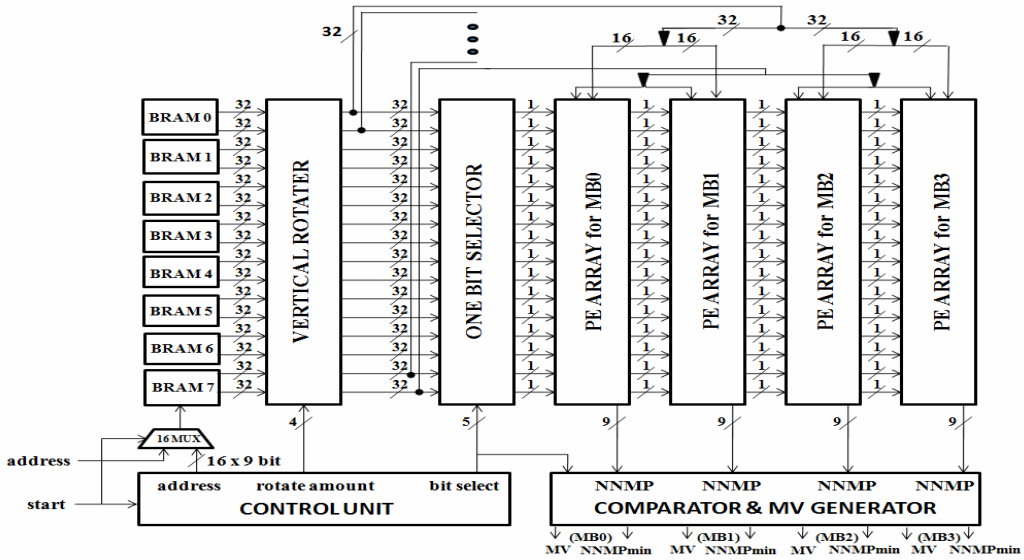


Fig. 1. Top-level Block Diagram of Proposed FBS ME Hardware

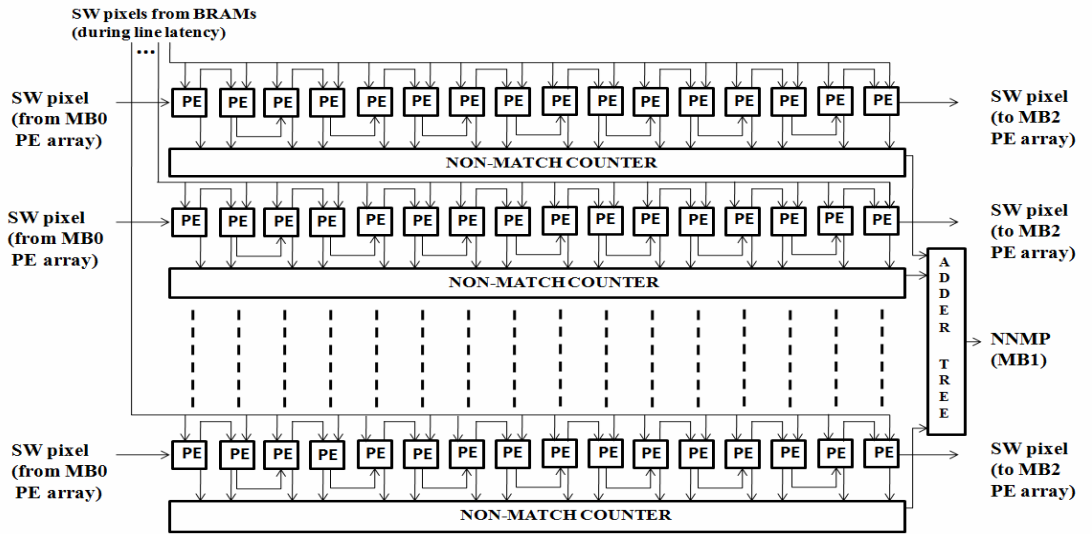


Fig. 3. MB1 PE Array for FBS ME Hardware

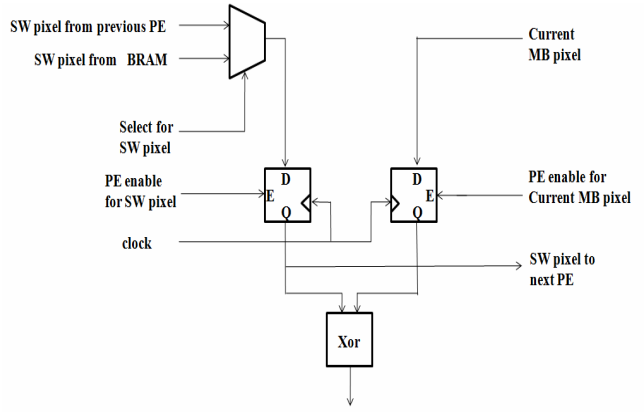


Fig. 4. PE Architecture

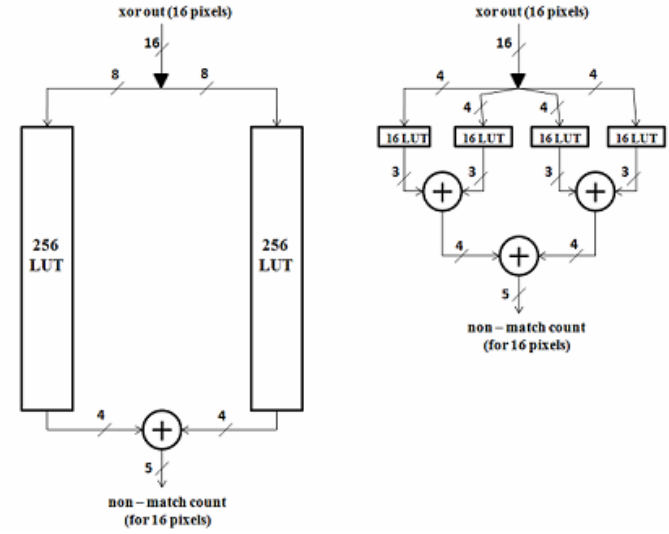


Fig. 5. Non-Match Counter Architectures
(a) Previous Architecture (b) Proposed Architecture

The architecture of a PE is shown in Fig. 4. Each PE performs an XOR operation between a SW pixel and a current MB pixel. The result of the XOR operation indicates whether these pixels match or not. The results of the XOR operations performed by all 256 PEs in a PE Array for a search location in the SW should be added to compute the NNMP value for that search location. In the proposed architecture, in order to compute the NNMP value for a search location, first, NNMP values for each row of the MB are computed by using Non-Match Counters, then the results of these 16 Non-Match Counters are added by an Adder Tree. Therefore, in the proposed hardware, there are 64 Non-Match Counters and 4 Adder Trees.

As shown in Fig. 5 (a), the Non-Match Counter used in the previous 1BT based ME hardware counts the ones in the outputs of 16 XOR gates by using 2 look up tables with 2^8 entries and adding the outputs of these look up tables. As shown in Fig. 5 (b), the Non-Match Counter we propose counts the ones in the outputs of 16 XOR gates by using 4 smaller look up tables with 2^4 entries and adding the outputs of these look up tables. The previous Non-Match Counter consumes 41 slices (82 LUTs) and has a 5.727ns delay. The proposed Non-Match Counter consumes 18 slices (35 LUTs) and has a 3.594ns delay. The proposed Non-Match Counter is faster and has smaller area. Since there are 64 Non-Match Counters in the proposed hardware, the proposed Non-Match Counter provides an area saving of (82-35)*64=3008 LUTs.

B. Memory Organization and Data Alignment

The memory organization of the 1BT based ME hardware architectures proposed in [12, 17] is shown in Fig. 6. These architectures have an inefficient memory organization. They implement full search algorithm for a [-16, 15] search range and a 16x16 MB size. This requires storing a 47x47 pixel = 2209 bits SW in on-chip memory. However, these architectures use 1504x16 = 24064 bits on-chip memory. Because they have pixel duplication in on-chip memory in order to be able to read 2x16 pixels from on-chip memory into PE array in each cycle. As it can be seen in Fig. 6, 15 pixels stored in addresses 0 and 47 are the same, and 15 pixels stored in addresses 47 and 94 are the same. Because of this memory organization, the amount of on-chip memory they use for

storing a 47x47 pixel SW is more than nine times the on-chip memory needed for storing a 47x47 pixel SW.

This inefficient memory organization also slows down the ME hardware proposed in [12, 17] because of high loading latency of the on-chip memory. These ME hardware compute the NNMP value for a search location in 1039 clock cycles. If 64 bits can be loaded into on-chip memory from off-chip memory in each clock cycle, 24064 bits on-chip memory for a search location can be loaded in 376 clock cycles and 256 bits current MB can be loaded in 8 clock cycles. This high loading latency of the on-chip memory reduces the performance of these ME hardware.

The memory organization of the 1BT based ME hardware proposed in this paper is shown in Fig. 7 and Fig. 8. 8 dual-port BRAMs in the FPGA are used to store the 48x96 SW_T. Region 0 includes first 16 lines of the SW_T, Region 1 includes lines 16 to 31 and Region 2 includes lines 32 to 47. Pixels in consecutive two lines of each region are stored in a BRAM. For example, first two lines of each region are stored in BRAM 0, and third and fourth lines of each region are stored in BRAM 1.

18 addresses of each BRAM are used and 32 bits are stored in each address. For example, pixels 0 to 31 in line 0 are stored in address 0 of BRAM 0, pixels 0 to 31 in line 1 are stored in address 1 of BRAM 0 and pixels 0 to 31 in line 2 are stored in address 0 of BRAM 1. Since SW of a single MB requires 2209 bits on-chip memory, SWs of 4 MBs requires 2209x4=8836 bits on-chip memory without data reuse. Because of the efficient data reuse scheme used in the proposed architecture, the proposed architecture uses 8x18x32 = 4608 bits on-chip memory for storing SWs of 4 MBs.

If 64 bits can be loaded into on-chip memory from off-chip memory in each clock cycle, the loading latency of the on-chip memory in the proposed ME hardware is 88 clock cycles; 72 clock cycles for loading 18 addresses of 8 BRAMs and 16 clock cycles for loading current MB pixels into PEs arrays. Therefore on the average 22 clock cycles loading latency is required for one MB which is much smaller than 384 clock cycles loading latency required by previous 1BT based ME hardware architectures.

8 BRAMs provide 16x32 bits in one clock cycle. Therefore, loading the necessary SW pixels for the first search location of a line from BRAMs into a PE array takes 2 clock cycles which is called *line latency*. For the search locations in the first line of SW_T, for all 8 BRAMs, Control Unit generates addresses 0 and 1 in the first clock cycle of *line latency*, addresses 2 and 3 in the second clock cycle of *line latency*, and addresses 4 and 5 in the following 32 clock cycles.

In the first clock cycle of *line latency*, 16x32 bits coming from Vertical Rotator is loaded into MB2 and MB3 PE arrays; the least significant 16 bits of each 32 bits are loaded into MB3 PE array and the most significant 16 bits of each 32 bits are loaded into MB2 PE array. In the second clock cycle of *line latency*, 16x32 bits coming from Vertical Rotator is loaded into MB0 and MB1 PE arrays; the least significant 16 bits of each 32 bits are loaded into MB1 PE array and the most significant 16 bits of each 32 bits are loaded into MB0 PE array. In the following 32 clock cycles after the *line latency*, in each clock cycle, 4 PE arrays compute NNMP values of 4 search locations in the same line.

Vertical Rotator is used to rotate the SW pixels read from the BRAMs for a search location in order to match them with the corresponding current MB pixels in the PE arrays. Vertical Rotator has 32 identical 16 bit rotators controlled by *rotate amount* signal. Since vertical rotation is not needed for the search locations in the first line of SW_T, *rotate amount* is 0 while computing the NNMP values of these search locations.

For the search locations in the second line of SW_T, in the first clock cycle of *line latency*, Control Unit generates addresses 1 and 6 for BRAM0 and addresses 0 and 1 for the other BRAMs. However, the SW pixels read from the address 1 of BRAM0 should be matched with the current MB pixels in the first row of MB2 and MB3, and the SW pixels read from the address 6 of BRAM0 should be matched with the current MB pixels in the 16^{th} row of MB2 and MB3. Therefore, vertical rotator is used to align the SW pixels with current MB pixels and *rotate amount* signal should be 1. In the second clock cycle of *line latency*, Control Unit generates addresses 3 and 8 for BRAM0 and addresses 2 and 3 for the other BRAMs. In the following 32 clock cycles, Control Unit generates addresses 5 and 10 for BRAM0 and addresses 4 and 5 for the other BRAMs. Therefore, data alignment is needed and *rotate amount* signal should be 1 for all the search locations in line 1 of SW_T.

0	Pixel 46:31 of line 0
1	Pixel 46:31 of line 1
2	Pixel 46:31 of line 2
⋮	⋮
46	Pixel 46:31 of line 46
47	Pixel 45:30 of line 0
48	Pixel 45:30 of line 1
49	Pixel 45:30 of line 2
⋮	⋮
94	Pixel 44:29 of line 0
95	Pixel 44:29 of line 1
96	Pixel 44:29 of line 2
⋮	⋮
1502	Pixel 15: 0 of line 45
1503	Pixel 15: 0 of line 46

Fig. 6. Memory Organization of Previous 1BT based ME Hardware

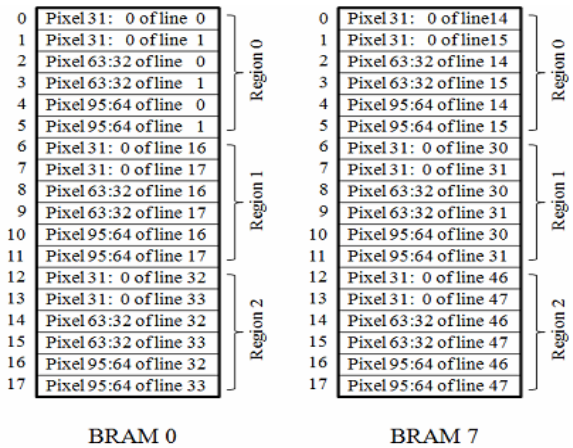


Fig. 7. Memory Organization of Proposed 1BT based ME Hardware

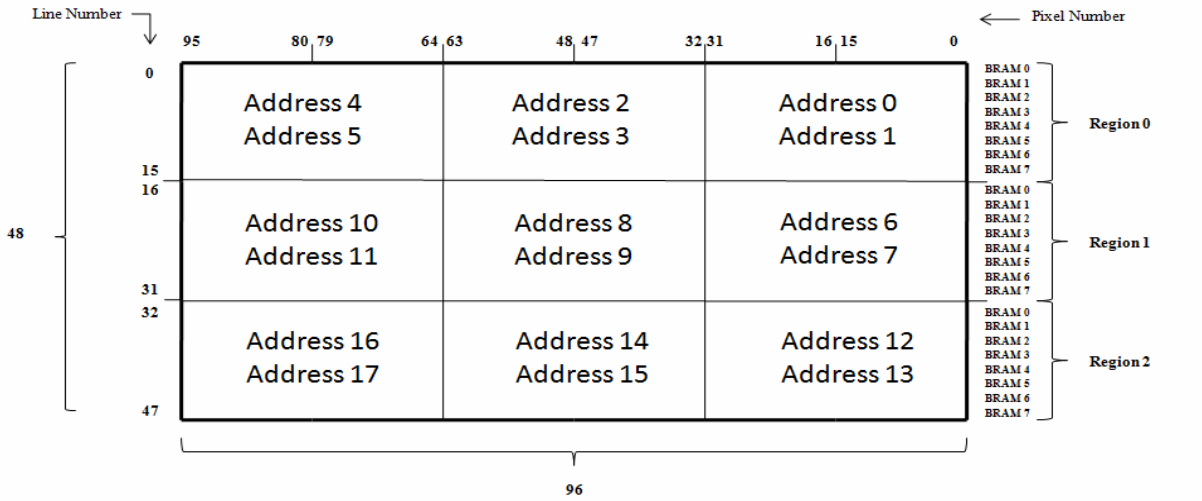


Fig. 8. Memory Organization of Proposed 1BT based ME Hardware

Since, for the search locations in the lines 16 and 32 of SW_T, SW pixels read from BRAMs and the corresponding current MB pixels in the PE arrays match without vertical rotation, while computing the NNMP values for the search locations in SW_T, *rotate amount* signal takes a value between 0 and 15.

One Bit Selector provides 16 new SW pixels to MB0 PE array for the remaining search locations in a line after the first search location in that line. One Bit Selector is controlled by the *bit select* signal. In the first clock cycle after the NNMP value for the first search location in a line is computed, *bit select* is 0 and the least significant 16 bits coming from vertical rotator are selected and these 16 pixels are sent to MB0 PE array. In the next clock cycle *bit select* is 1 and the second 16 bits coming from vertical rotator are selected and these 16 pixels are sent to MB0 PE array. In this way, *bit select* signal counts from 0 to 31 and in each clock cycle the corresponding 16 new SW pixels are sent to the MB0 PE array. In the last clock cycle *bit select* is 31 and the most significant 16 bits coming from vertical rotator are selected and these 16 pixels are sent to MB0 PE array.

IV. Proposed Hardware Architecture For 1BT Based Variable Block Size Motion Estimation

The top-level block diagram of the proposed VBS ME hardware architecture is similar to the top-level block diagram of the proposed FBS ME hardware architecture shown in Fig. 1. The Non-Match Counters (NMC) and Adder Trees used in the PE arrays in FBS ME hardware and the ones used in the PE arrays in VBS ME hardware are different. MB1 PE array for VBS ME hardware is shown in Fig. 9. As shown in Fig. 3 and Fig. 9, even though each PE array in FBS ME hardware computes the NNMP value for a MB, each PE array in VBS ME hardware computes the NNMP values for the 41 partitions of a MB. The 41 partitions of a MB are shown in Fig. 10.

In VBS ME hardware, a NMC in a PE array computes the NNMP value for 4 current MB and 4 SW pixels using a look up table with 2^4 entries. In the Adder Tree, the outputs of the NMCs are added to compute the NNMP values for the 16 4x4 blocks. For example, NMC (0, 0), NMC (1, 0), NMC (2, 0) and NMC (3, 0) are added to compute the NNMP value for 4x4 block 1 as shown in Fig. 10. The NNMP values for the 4x4 blocks are added to compute the NNMP values for the 4x8 and 8x4 blocks and these NNMP values are stored in pipeline registers.

The NNMP values for the 4x8 blocks are added to compute the NNMP values for the 8x8 blocks. The NNMP values for the 8x8 blocks are added to compute the NNMP values for the 8x16 and 16x8 blocks and these NNMP values are stored in pipeline registers. The NNMP values for the 8x16 blocks are added to compute the NNMP value for the 16x16 block. Therefore, the pipelining in the Adder Trees in VBS ME hardware causes 2 clock cycles latency for computing the NNMP value for a 16x16 MB same as the Adder Trees in FBS ME hardware.

The 41 NNMP values of each MB computed by a PE array are sent to Comparator & MV Generator, and the Comparator & MV Generator determines the minimum NNMP values and the corresponding MVs for each MB partition.

V. Implementation Results

The proposed 1BT based FBS ME and VBS ME hardware architectures are implemented in Verilog HDL. The Verilog RTL codes are synthesized with Mentor Graphics Precision RTL 2005b and mapped to a Xilinx XC2VP30-7 FPGA using Xilinx ISE 8.2i. The hardware implementations are verified with post place & route simulations using Mentor Graphics Modelsim 6.1c.

The FBS ME hardware consumes 4758 slices (7280 LUTs), which is %34 of all the slices of a XC2VP30-7 FPGA. A PE array consumes 547 slices (1094 LUTs), Vertical Rotator consumes 1024 slices (2048 LUTs), One Bit Selector consumes 128 slices (256 LUTs) and the remaining slices are used for Comparator & MV Generator, Control Unit and multiplexers before address ports of the BRAMs. In addition, 4608 bits on-chip memory is used for storing SWs of 4 MBs, and these 4608 bits are stored in 18 addresses of 8 BRAMs.

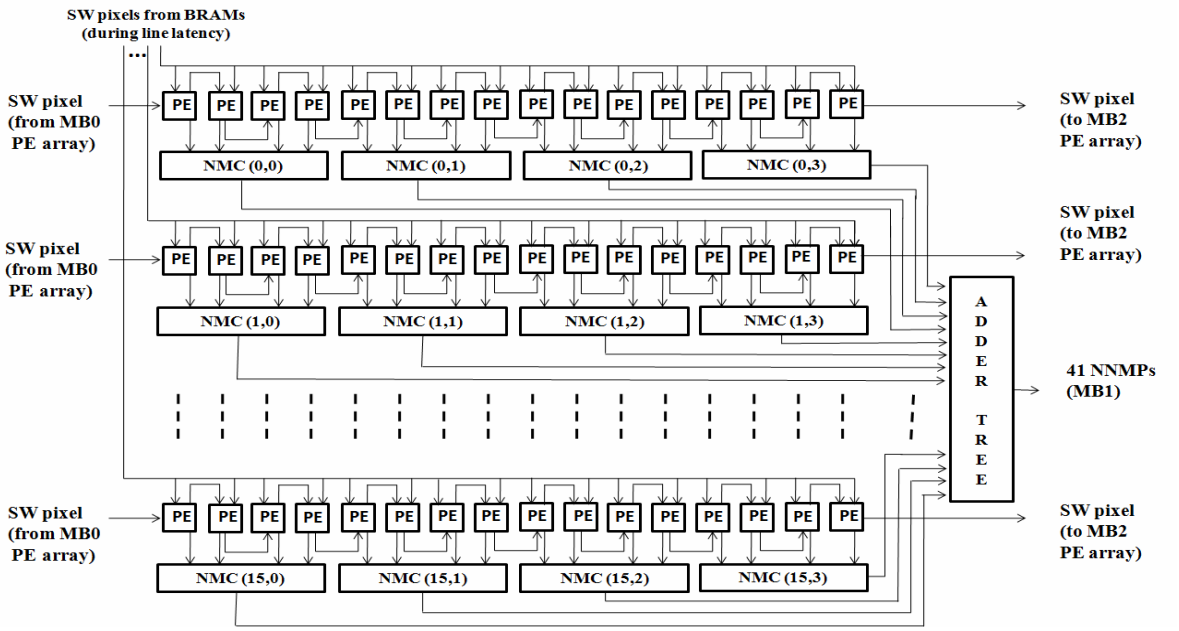


Fig. 9. MB1 PE Array for VBS ME Hardware

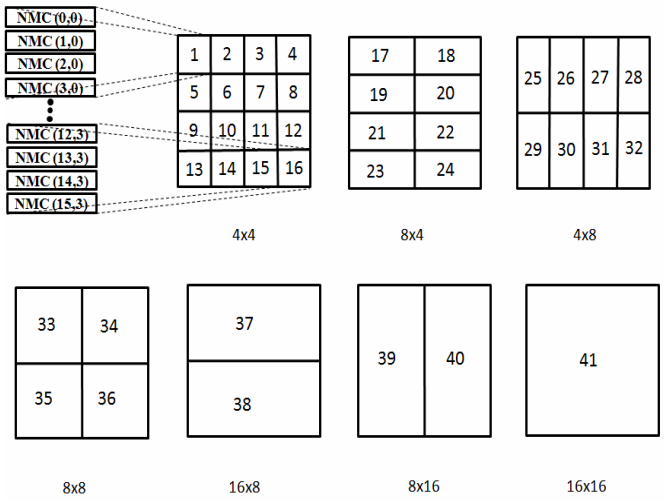


Fig. 10. Macroblock Partitions

The VBS ME hardware consumes 6782 slices (8702 LUTs) which is %49 of all the slices of a XC2VP30-7 FPGA. Since VBS ME hardware has more complex Comparator & MV Generator and Adder Tree than FBS ME hardware, it consumes more LUTs and DFFs than FBS ME hardware.

For both FBS ME and VBS ME hardware, starting the search in a line has a 2 clock cycles *line latency*. Because of the [-16, 16] search range, there are 33 lines in a SW and 33 search locations in each line are searched. 6 stage pipelining causes 6 clock cycles latency. Therefore, ((32+2) x 33) + 6 = 1128 clock cycles are required by both ME hardware for processing 4 MBs and on the average processing one MB requires 282 clock cycles. Both ME hardware can work at 113 MHz after place & route. Therefore, they are capable of processing 49 1920x1080 full HD frames per second.

The 1BT based ME hardware architectures proposed in this paper are based on the ME hardware architecture proposed in [11]. However, the ME hardware proposed in [11] is performing 8 bits/pixel ME using SAD block matching criterion, it is implementing a Hexagon-Based ME algorithm, it is not performing ME for 4 MBs in parallel, and it is not performing VBS ME.

The comparison of the proposed FBS ME and VBS ME hardware with the Full Search ME hardware proposed in [4, 5, 12, 17] is shown in Table I. The proposed 1BT based ME hardware architectures are faster and have less logic area and on-chip memory than the 8 bits/pixel VBS ME hardware architectures proposed in [4, 5].

We synthesized the 1BT based ME hardware architectures presented in [12, 17] using Mentor Graphics Precision RTL 2005b and mapped them to a Xilinx XC2VP30-7 FPGA using Xilinx ISE 8.2i. The 1BT ME hardware architecture proposed in [12] consumes 998 Slices (1589 LUTs) and 24064 bits on-chip memory for storing the search window. It requires 1039 clock cycles for processing one MB for a [-16, 15] search range. It works at 117 MHz and it can process 13 1920x1080 full HD frames per second. The MF1BT ME hardware architecture proposed in [17] consumes 944 Slices (1467 LUTs) and 24064 bits on-chip memory for storing the search window. It requires 1039 clock cycles for processing one MB for a [-16, 15] search range. It works at 127 MHz and it can process 15 1920x1080 full HD frames per second.

The area of the proposed 1BT based ME hardware architectures are larger than the area of the 1BT based ME hardware architectures proposed in [12, 17] because of performing ME for 4 MBs in parallel and data alignment. However, the proposed ME hardware architectures are much faster and have much less on-chip memory than these ME hardware architectures.

TABLE I
COMPARISON OF MOTION ESTIMATION HARDWARE ARCHITECTURES

	Proposed (FBS)	Proposed (VBS)	[12]	[17]	[4]	[5]
Bit Depth	1	1	1	1	8	8
On-Chip SW Memory (bits)	4608	4608	24064	24064	26624	24192
Area	7280 LUTs 2745 DFFs	8702 LUTs 6401 DFFs	1589 LUTs 478 DFFs	1467 LUTs 499 DFFs	160K Gates	76400 LUTs 18000 DFFs
Maximum Frequency (MHz)	115	113	117	127	200	198
Technology	XC2VP30 FPGA	XC2VP30 FPGA	XC2VP30 FPGA	XC2VP30 FPGA	0.18µm ASIC	XC5VLX330 FPGA
Search Range	[-16, 16]	[-16, 16]	[-16, 15]	[-16, 15]	[-16, 16]	[±24, ±16]
Search locations / MB	1089	1089	1024	1024	1089	1584
Performance (1920x1080 fps)	50	49	13	15	21	31
Performance (1280x720 fps)	113	111	31	33	49	69
Supported MB Partitions	16x16	4x4, 4x8, 8x4, 8x8, 16x8, 8x16, 16x16	16x16	16x16	4x4, 4x8, 8x4, 8x8, 16x8, 8x16, 16x16	4x4, 4x8, 8x4, 8x8, 16x8, 8x16, 16x16

VI. CONCLUSION

In this paper, we presented high performance systolic hardware architectures for 1BT based FBS ME and VBS ME. The proposed hardware architectures perform full search ME for 4 MBs in parallel and they are faster than the 1BT based ME hardware reported in the literature. They use less on-chip memory than the previous 1BT based ME hardware by using a novel data reuse scheme and memory organization. The proposed VBS ME hardware is the first 1BT based ME hardware implementing VBS ME. The FBS ME and VBS ME hardware consume %34 and %49 of the slices in a Xilinx XC2VP30-7 FPGA, respectively. They can work at 113 MHz in the same FPGA and are capable of processing 49 1920x1080 full HD frames per second. Therefore, they can be used in consumer electronics products that require real-time video processing or compression.

REFERENCES

[1] I. Richardson, *H.264 and MPEG-4 Video Compression,* Wiley, 2003.

[2] B.-D. Choi, J.-W. Han, C.-S. Kim, S.-J. Ko, "Motion-compensated Frame Interpolation Using Bilateral Motion Estimation and Adaptive Overlapped Block Motion Compensation," *IEEE Trans. on Circuits Syst. Video Technol.*, vol. 17, no.4, pp. 407–416, Apr. 2007.

[3] Y. Ling, J. Wang, Y. Liu, and W. Zhang, "A Novel Spatial and Temporal Correlation Integrated Based Motion-compensated Interpolation for Frame Rate Up-conversion," *IEEE Trans. on Consumer Electron.*, vol. 54, no.2, pp. 863-869, May 2008.

[4] C. Wei, H. Hui, T. Jiarong, and M. Hao, "A High-performance Reconfigurable VLSI Architecture for VBSME in H.264," *IEEE Trans. on Consumer Electron.*, vol. 54, no. 3, pp. 1338-1345, Aug. 2008.

[5] T. Moorthy, and A. Ye, "A Scalable Computing and Memory Architecture for Variable Block Size Motion Estimation on Field-Programmable Gate Arrays," *International Conference on Field Programmable Logic*, pp. 83-88, Sept. 2008.

[6] R. Li, B. Zeng, and M.L. Liou, "A New Three-step Search Algorithm for Block Motion Estimation," *IEEE Trans. on Circuits Syst. Video Technol.*, vol. 4, pp. 438–442, 1994.

[7] S. Zhu and K.-K. Ma, "A New Diamond Search Algorithm for Fast Block Matching Motion Estimation," *IEEE Trans. on Image Processing*, vol. 9, pp. 287–290, 2000.

[8] C. Zhu, X. Lin, and L. P. Chau, "Hexagon-based Search Pattern for Fast Block Motion Estimation," *IEEE Trans. on Circuits Syst. Video Technol.*, vol. 12, pp. 349–355, 2002.

[9] X.-Q. Banh and Y.-P. Tan, "Adaptive Dual-cross Search Algorithm for Block-matching Motion Estimation", *IEEE Trans. on Consumer Electron.*, vol. 50, no. 2, pp. 766-775, May 2004.

[10] W. M. Chao, C. W. Hsu, Y. C. Chang, and L. G. Chen, "A Novel Motion Estimator Supporting Diamond Search and Fast Full Search," *IEEE ISCAS*, May 2002.

[11] O. Tasdizen, A. Akin, H. Kukner, I. Hamzaoglu, and H. F. Ugurdag, "High Performance Hardware Architectures for a Hexagon-based Motion Estimation Algorithm," *16th IEEE / IFIP International Conference on VLSI - SoC*, Rhodes Island, Greece, Oct. 2008.

[12] B. Natarajan, V. Bhaskaran, K. Konstantinides, "Low-complexity Block-based Motion Estimation via One-bit Transforms," *IEEE Trans. on Circuits Syst. Video Technol.,* vol. 7, no. 3, pp. 702-706, Aug 1997.

[13] S. Ertürk, "Multiplication-Free One-Bit Transform for Low-Complexity Block-Based Motion Estimation," *IEEE Signal Process. Letters*, vol. 14, no. 2, pp. 109-112, Feb. 2007.

[14] H. Lee and J. Jeong, "Early Termination Scheme for Binary Block Motion Estimation," *IEEE Trans. on Consumer Electron*., vol. 53, no. 4, pp. 1682-1686, Nov. 2007.

[15] A. Ertürk, S. Ertürk, "Two-Bit Transform for Binary Block Motion Estimation," *IEEE Trans. on Circuits Syst. Video Technol*., vol. 15, no. 7, pp. 938-946, July 2005.

[16] O. Urhan and S. Ertürk, "Constrained One-Bit Transform for Low Complexity Block Motion Estimation," *IEEE Trans.on Circuits Syst. Video Technol.*, vol. 17, no. 4, pp. 478-482, Apr. 2007.

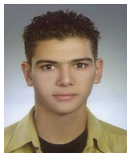
[17] A. Çelebi, O. Urhan, İ. Hamzaoğlu, S. Ertürk, "Efficient Hardware Implementations of Low Bit Depth Motion Estimation Algorithms," *IEEE Signal Process. Letters*, 2009.

[18] A. Çelebi, O. Akbulut, O. Urhan, İ. Hamzaoğlu, S. Ertürk, "An All Binary Sub-Pixel Motion Estimation Approach and its Hardware Architecture," *IEEE Trans. on Consumer Electron*., vol. 54, no.4, Nov. 2008.

[19] J.-C. Tuan, T.-S. Chang, and C.-W. Jen, "On the Data Reuse and Memory Bandwidth Analysis for Full-search Block-matching VLSI Architecture", *IEEE Trans. on Circuits Syst. Video Technol.*, vol. 12, pp. 61-72, 2002.

[20] S. Goel, Y. Ismail, P. Devulapalli, J. McNeely, M. Bayoumi, "An Efficient Data Reuse Motion Estimation Engine", *IEEE Signal Process. Syst. Design and Imp.*, pp. 383-386, Oct. 2006.

Abdulkadir Akın received B.S degree in Electronics Engineering from Sabanci University, Istanbul, Turkey in July 2008. He is currently studying towards an M.S. degree in Electronics Engineering at Sabanci University. His research interests include digital hardware design for video compression and video enhancement.



Yigit Dogan is currently studying towards B.S degree in Electronics Engineering at Sabanci University, Istanbul, Turkey. His research interests include digital hardware design for video compression and video enhancement.

İlker Hamzaoğlu (M'00) received B.S. and M.S. degrees in Computer Engineering from Bogazici University, Istanbul, Turkey in 1991 and 1993 respectively. He received Ph.D. degree in Computer Science from University of Illinois at Urbana-Champaign, IL, USA in 1999. He worked as a Senior and Principle Staff Engineer at Multimedia Architecture Lab, Motorola Inc. in Schaumburg, IL, USA between August 1999 and August 2003. He is working as an Assistant Professor at Sabanci University, Istanbul, Turkey since September 2003. His research interests include SoC ASIC and FPGA design for digital image and video processing and coding, low power digital SoC design, digital SoC verification and testing.