IJCAES
ISSN: 2231-4946
Volume III, Special Issue, August 2013

International Journal of Computer Applications in Engineering Sciences
Special Issue on National Conference on Information and Communication (NCIC'13)
www.caesjournals.org

Area Efficient SAD Architecture for Block Based Video Compression Standards

S. Archana[1], D. Rukmani Devi[2]

[1]*M.E VLSI DESIGN, RMK Engineering College*
Chennai, India.

[1]archansundar@gmail.com

***Abstract*—Block based motion estimation is one of the critical task in video compression standards such as MPEG-4, H.263, H.264. The key element of the block based motion estimation algorithms is the matching criteria. SAD(Sum of Absolute Difference) is the most common matching criteria chosen in video coding because of its low complexity, good performance and ease of hardware implementation. By utilizing the concept of reversibility a novel compression array unit using reversible 4:2 compressor has been proposed. Experimental results indicate that the proposed compression unit will impose effectiveness on the SAD architecture in calculating the motion vectors with reasonable area overhead and power penalty.**

***Keywords*—Motion estimation, Sum of absolute difference, 4:2 Compressor, SAD architecture, Reversible gates.**

I. INTRODUCTION

Motion estimation (ME) plays an important role in video coding and processing systems since motion vectors are critical information for temporal redundancy reduction. It has been widely employed in the MPEG-4; H.263 and H.264/MPEG-4 AVC block based video compression standards. Motion estimation is defined as searching the best motion vector which is the displacement of the coordinate of the best similar block in previous frame for the block in current frame.

Block-based matching method is the most widely used motion estimation method for video compression since pictures are normally rectangular in shape and block-division can be easily done. Usually, standards bodies, e.g. MPEG, define the standard block sizes for motion estimation. This can be 16 by 16, 8 by 8, etc, depending on the target application of the video codec. The heavy computational cost of the block matching algorithms (BMA) can be a significant problem in real time coding applications. It obtains the best match by minimizing a cost function. Various matching criteria have been proposed and analyzed in the literatures, varying in complexity and efficiency. In this section, the mean absolute difference (MAD), mean square error (MSE), sum of absolute difference (SAD) and sum of absolute transformed difference (SATD) block matching criteria are explained[1].

Though MAD is simple and easy for implementation in hardware, unfortunately MAD tends to overemphasize small differences, giving an inferior result to MSE. MSE is accurate but its complexity is high for both software and hardware implementations. SATD, although it offers significant improvement in prediction quality, transform hardware must be added within the motion estimation loop and hardware complexity is increased. On the other hand, the latency and thus the performance of motion estimation will be degraded due to the added transform hardware. This technique is applied to H.264/AVC when performing quarter pixel accuracy motion estimation. The high accuracy is needed since it is the final stage of motion estimation.

SAD is the most common matching criteria chosen in video coding because of its low complexity, good performance and ease of hardware implementation. The only difference between SAD and MAD is that SAD takes the sum of all pixels while MAD measures the average pixel value. Since the block size is constant during

subtraction, the average value per pixel is not important. A divide operation is saved and the overall computation is simplified.

II. SAD ARCHITECTURE

SAD is an extremely fast metric due to its simplicity; it is effectively the simplest possible metric that takes into account every pixel in a block. Therefore it is very effective for a wide motion search of many different blocks. SAD is also easily parallelizable since it analyzes each pixel separately, making it easily implementable. Consider the block size of a motion picture be M×N, the location of the current block is (x, y) and the corresponding motion vector (MV) is (i, j). Then the SAD for the current block is defined as

$$SAD(x,y,i,j) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\left|A_{(x+u,y+v)} - B_{(x+i+u,y+j+v)}\right|$$

The SAD architecture can be divided into three stages: absolute differences calculation, absolute differences accumulation, and minimum SAD determination. Unlike many hardware implementations based on complement adders, Instead of using the area-consuming adders, the *LiYufei's* [6] SAD architecture is implemented based on the comparison of unsigned numbers. Comparing to the reference architectures, the *Li Yufei's* architecture reduces the area significantly at a cost of negligible latency. *LiYufei* has implemented the compression array unit using 4:2 compressor made of pass transistor multiplexer. In this paper an attempt to modify the compression array unit using reversible gates has been made. The whole SAD architecture is shown in figure 1.

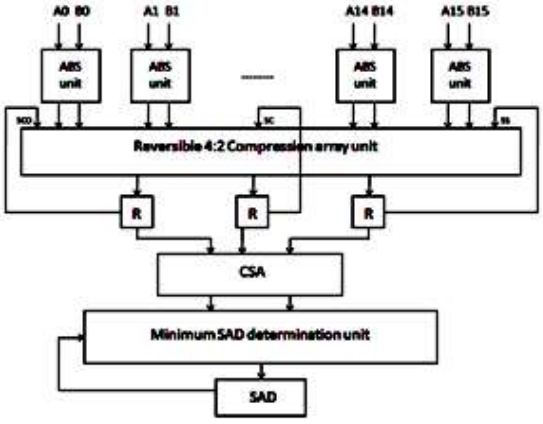

Fig. 1 SAD Architecture

A. *Absolute Difference Calculation Unit*

The Absolute difference unit calculates the absolute differences between each pixel in the current block and the corresponding pixel in the block being compared. This unit implemented using comparators. In this paper the one and two bit comparators are referred as comp-1 and comp-2. Figure 2 is the one bit comparator comp-1 and two bit comparator comp-2 is shown in figure 3. Figure 4 is the whole absolute difference unit. The implementation using comparators has a reasonable area compared to the architectures that uses adders.

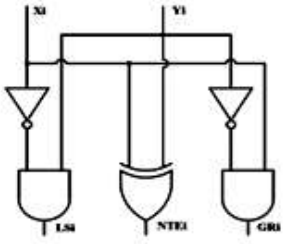

Fig.2 COMP-1

The X and Y in [6] are implemented by X XOR GR and Y XOR LS signals respectively. The LS, GR and NTE represent the less, greater and not equal signals. In the compression array unit, the compression array is used to calculate the 16 groups of partial results composed of {a7, a6... a0}, {b7, b6…b0} and NTE.

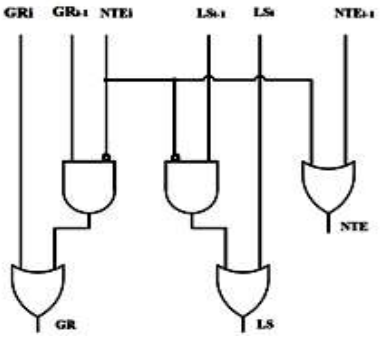

Fig. 3 COMP-2

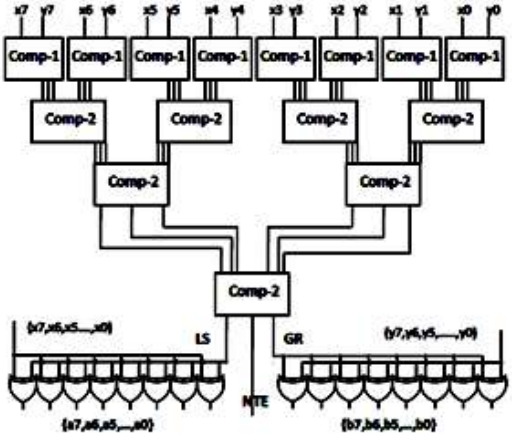

Fig. 4 Absolute Difference Calculation Unit

B. B. Proposed Compression Array Unit

All of the information encoded in the input must be preserved at the output in computational tasks such as digital signal processing, communication, computer graphics and cryptography. The loss of information is associated with laws of physics requiring that one bit of information lost dissipates k T ln 2 of energy, where k is Boltzmann's constant and T is the temperature of the system. Interest in reversible computation arises from the desire to reduce heat dissipation, thereby allowing higher densities and higher speed. Hence there are compelling reasons to consider circuits composed of reversible gates and the synthesis of such networks.

Reversible circuits are circuits (gates) that have one to-one mapping between vectors of inputs and outputs; thus the vector of input states can be always reconstructed from the vector of output states.

In this paper, a novel compression array unit has been designed using reversible 4:2 compressor from the DKGP gates. It has been proved that the SAD architecture proposed is reasonable than its counterpart existing in literature, in terms of power and area. This is the first attempt to design a compression array unit using a reversible 4:2 compressor as far as existing literature and our knowledge is concerned.

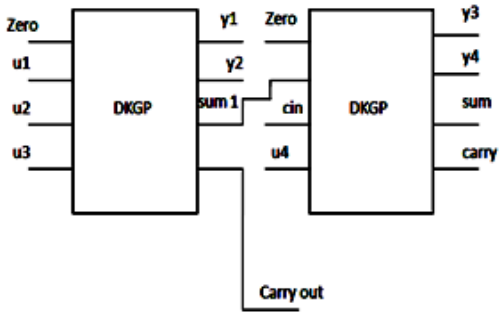

Fig. 5 Reversible 4:2 Compressor designed from DKGP gates

The 4:2 compressor implemented using DKGP gates in [4] has been utilized to design the compression array unit. The four input bits which are given as input to the 4:2 compressor are compressed to two bits. The delay of 4:2 compressor is 3 XOR gates in series. The reversible 4:2 compressor designed from two DKGP gates and its block diagram are shown in Figure 5 and 6 respectively.

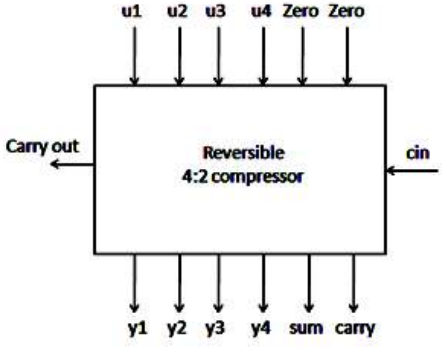

Fig. 6 Block diagram of Reversible 4:2 Compressor

The input of the compression array unit is composed of 16 groups of {a7,a6,a5,…,0}, {b7,b6,b5,…,0} and 3 recursive results SS, SC and SCO. Totally there are 35 groups of input. An AND operation is performed between an int signal and the partial results to initialize the recursive values. The whole compression array unit is shown in figure 8. The sixteen NTEs are compressed separately and the results are inserted into the compression tree. The NTE compression and initialization are shown in figure 7. In the compression tree the partial results are segregated as MSB and LSB and inserted into the tree correspondingly. The result of the compression array unit is carried over to the minimum SAD determination unit.

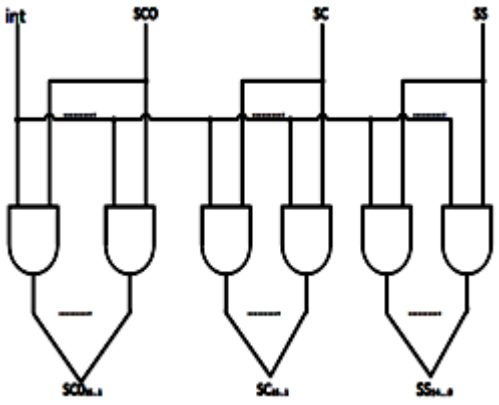

Fig. 7 AND array initialization

For simplicity the sum, carry and carry out are only shown in the figure, the other outputs are not shown in the figure 7 and 8.

Fig. 7 NTE compression

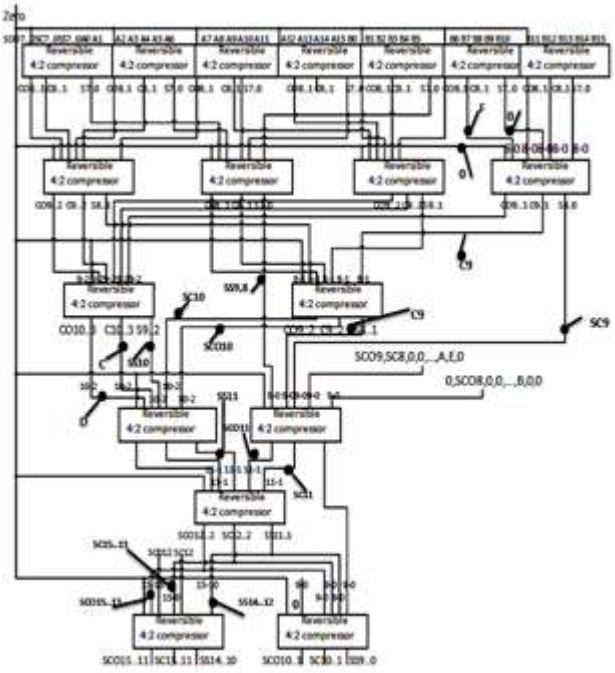

Fig. 8 Reversible Compression array unit

C. Minimum Sad Determination Unit

The minimum SAD determination unit is composed of a 16-bit carry look ahead adder, 16 bit comparator and a multiplexer. The 16 bit comparator is composed of one and two bit less compressor. In this paper one and two bit less compressor are represented as cmpsr-1 and cmpsr-2. It is shown is figure 9 and 10 respectively.

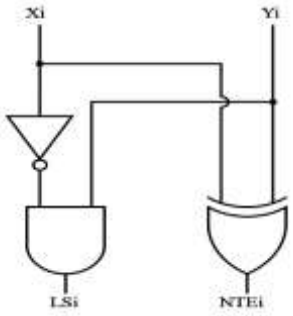

Fig. 9 Cmpsr-1

The cmpsr-2 is composed of half compressor (HC) and an AND gate. The comparator operates in parallel. The 16 bit comparator and the whole minimum SAD unit are shown in figure 11 and 12 respectively.

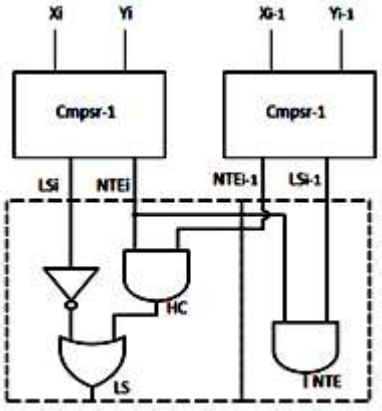

Fig. 10 Cmpsr-2

The area cost is reduced significantly due to the area optimization. LS signal denotes that the candidate SAD is less than the current SAD. If LS equals 1 the current SAD is replaced by the candidate.

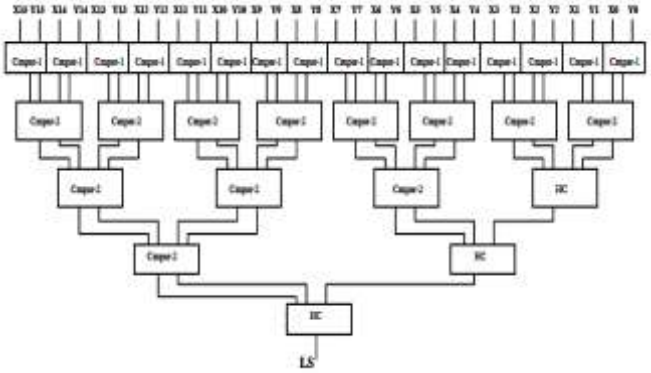

Fig. 11 16 bit comparator

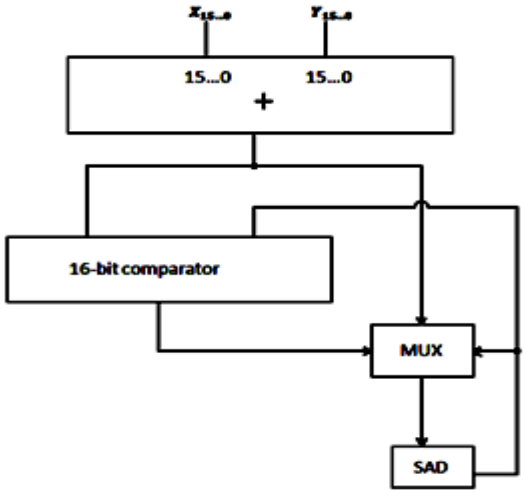

Fig.12 Minimum SAD determination unit

III. RESULTS AND DISCUSSION

The proposed compression array unit using reversible 4:2 compressor is simulated using Xilinx ISE simulator and the area and power are calculated using Design Vision. The RTL schematic view of the proposed compression array unit is shown is figure 15. The output of the absolute difference unit and the recursive results are given as input to the compression array unit and area and power are calculated. The design vision reported the area and power as 7398 units and 5.1mW. The Area and power reports of the proposed compression array unit are shown in the figure 13 and 14.

```
design_vision> report_area

****************************************
Report : area
Design : cau
Version: G-2012.06
Date   : Sat Mar  9 14:17:50 2013
****************************************

Library(s) Used:

    fsd0a_a_generic_core_tt1v25c (File: /opt/techlibs/fsd0a_a/2011Q2v2.2/GENERIC
_CORE_1D0V/FrontEnd/synopsys/synthesis/fsd0a_a_generic_core_tt1v25c.db)

Number of ports:                          381
Number of nets:                           773
Number of cells:                           19
Number of combinational cells:              0
Number of sequential cells:                 0
Number of macros:                           0
Number of buf/inv:                          0
Number of references:                      19

Combinational area:       7398.688142
Noncombinational area:       0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)
```

Fig. 13 Area report

```
  Cell Internal Power  =   3.8785 mW   (59%)
  Net Switching Power  =   2.1707 mW   (41%)
                         ---------
Total Dynamic Power    =   5.2493 mW  (100%)

Cell Leakage Power     =  17.1517 uW

Information: report_power power group summary does not include estimated clock t
ree power. (PWR-789)

                 Internal         Switching          Leakage            Total
Power Group      Power            Power              Power              Power
(   %    )  Attrs
--------------------------------------------------------------------------------
io_pad           0.0000           0.0000             0.0000             0.0000
(    0.00%)
memory           0.0000           0.0000             0.0000             0.0000
(    0.00%)
black_box        0.0000           0.0000             0.0000             0.0000
(    0.00%)
clock_network    0.0000           0.0000             0.0000             0.0000
(    0.00%)
register         0.0000           0.0000             0.0000             0.0000
(    0.00%)
sequential       0.0000           0.0000             0.0000             0.0000
(    0.00%)
combinational    3.8785           2.0843         1.7152e+07             5.1800
(  100.00%)
--------------------------------------------------------------------------------
Total            3.8785 mW        2.0843 mW      1.7152e+07 pW          5.1800
mW
```

Fig. 14 Power report

Fig. 15 RTL Schematic of proposed Compression Array unit

IV. CONCLUSION

This paper presents the SAD architecture used to in estimation of motion vectors in the block based video compression standards. By utilizing the concept of reversibility a novel compression array unit using reversible 4:2 compressor has been proposed. Experimental results indicate that the proposed compression unit will impose effectiveness on the SAD architecture in calculating the motion vectors with reasonable area overhead and power penalty.

REFERENCES

[1] Chen H and Shu Q, *Apparatus for implementing a block matching algorithm for motion estimation in video image processing*, U.S. Patent 5 864 372, Jan. 26, 1999.

[2] Jarno Vanne, Ewro Aho, Timo D. Hämäläinen and Kimmo kuusilinna, *A High-Performance Sum of Absolute Difference Implementation for Motion Estimation*, IEEE Trans. on Circuits and Systems for Video Technology, Vol. 16, no. 7, July, 2006.

[3] Jheng Y.S, Chen L.G and Chiueh T.D, *An efficient and simple VLSI tree architecture for motion estimation algorithms*, IEE Trans. On Signal Processing, Vol. 42, no. 2, Feb. 1993.

[4] D. Krishnaveni, M. Geetha Priya and K. Baskaran, *Design of an Efficient Reversible 8x8 Wallace Tree Multiplier*, World Applied Sciences Journal 20(8):1159-1165, ISSN 1818-4952, 2012.

[5] Ng Ka Ho (2008), *High Efficient Motion Estimation Algorithms in Video Coding*, City University of Hong Kong, March, 2008.

[6] LiYufei, FengXiubo and WangQin (2007), *A High Performance Low Cost SAD Architecture for Video Coding*, IEEE Trans. On Consumer Electronics, Vol. 53, no. 2, MAY, 2007.

[7] Thomas Wiegand Sullivan G.J, Gisle Bjøntegaard, and Ajay Luthra, *Overview of the H.264/AVC Video Coding Standard*, IEEE Trans. On Circuits and Systems for video technology, vo. 13, no.7, July, 2003.