Implementation of SAD Architecture for Motion Estimation in H.264/AVC

R.Mohamed Niyas [#1], M.Guru [#2], P.Jayakrishnan [*3]
[#] VLSI Design-SENSE Department, VIT University
Vellore India
[1] mdnys@yahoo.com
[2] mguru.21@gmail.com
[*] Professor, VIT University
Vellore India
[3] pjayakrishnan@vit.ac.in

***Abstract*— The last decade has seen a quiet uprising in digital video technology. The digital videos are been used in television, broadcasting, Internet and also in mobile phones. The uprising issues use H.264/AVC formats in video compression techniques. The SAD operation can be divided into two stages, viz. absolute and sum stage. Pipelining method is implemented as it is used in low power applications like mobile devices. For checking the functionality, HDL verification is done by using ModelSim Simulator. This paper represents the idea of yielding a faster outcome than real-time Full-Search (FS) in Motion Estimation (ME) for the main profile/main level of the H.264 standard. In order to generate the outcomes, we use TSMC 45 nm standard-cells library by using Cadence RTL Compiler tool and the obtained frequency of about 176.05 MHz.**

Keyword- Hardware Description Language (HDL), Sum of Absolute Difference (SAD), Advanced Video Coding (AVC).

I. INTRODUCTION

Now-a-days video compression methods are used in many marketable products, from user electronic devices namely, cameras, mobile handsets to video-chatting systems. To increase the performance of the current uses and to enable the applicability of video compression to new real-time uses, recently, a novel standard has been established for video compression. This offers significantly improved video compression competence than aforementioned video compression standards, is developed with the collaboration of ITU and ISO standardization societies. Hence it is called with two distinctive terms, namely, MPEG4 Part-10 and H.264/AVC.

The effectiveness attained in H.264 standard is not an outcome of any single feature but the combination of numerous encoding tools in the compression of video techniques. The top-level block of an H.264 Encoder consists of motion compression, entropy coding, transformations and ME. From this one of these tools is the block size ME used in the baseline profile of H.264AVC standard. The ME is one of the most demanding parts of the encoders fulfilling the previous video compression touchstones. In this paper fixed block size ME achieves better coding outcomes compared to the previous video compression standards. Hence, this coding comes with a rise in complexity of encoding that makes it an challenge to have a real-time implementation of ME for H.264 video coding.

In this paper, we present a high performance and low cost hardware architecture for real-time implementation of an SAD reuse based hierarchical ME algorithm for H.264/AVC. The proposed architecture is implemented in Verilog HDL. The new design for real-time implementation of a fixed block size ME algorithm for H.264 video coding is presented in this paper. This design attains greater performance and also reduction in number of gate counts have been obtained.

The reminder of the paper are as follows, Section II conveys the outline of ME architectures and a new design of SAD architecture which is developed for fixed block size ME. Section III conveys detailed explanation about the SAD implementation. The block diagram and outcomes of the SAD architecture is provided in Section IV. Finally, Section V comprises the conclusion of the paper.

II. BACKGROUND

ME methods form the core of video compression and video processing applications. It obtains motion information from the video sequence. The motion is typically represented using a Motion Vector (MV). The MV denotes the displacement of a pixel or a pixel block from the current location due to motion. It is used in applications like motion video stabilisation; motion tracking etc. there are two varieties of block matching techniques are available namely, fixed and variable block sizes. In this paper, a new SAD architecture has been designed using fixed block sizes. This paper also reviews the technique of Block Matching Algorithm (BMA).

This is the most popular ME algorithm and it calculates MV for an entire block of pixels instead of individual pixels. The same MV is applicable to all the pixels in the block. This inturn reduces the computational requirement and also outcomes in accurate MV since the objects are typically in a group of pixels. BMA algorithm is shown in fig 1. The current frame is divided into pixel blocks and motion estimation is performed independently for each pixel block. ME is done by finding a pixel block from the reference frame that best matches the current block, whose motion is being estimated. SAD can be calculated by the given equation (1) as follows.

Sum of Absolute Difference, $$SAD = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left| C(x,y) - R(x,y) \right| \quad (1)$$

In the above equations, C(x,y) and R(x,y) denotes the current pixel area and the reference pixel area. SAD delivers good match at lesser computational requirements. Hence it is usually used for block matching techniques. The reference pixel blocks are generated only from a region known as the search area. Search region defines the boundary for the motion vectors and limits the number of blocks to evaluate. The search area can be improved depending upon the detected motion. The horizontal and vertical search range, Sx & Sy, define the search area (+/-Sx and +/- Sy) as illustrated in fig 1. (i.e. the position of the candidate blocks within the search region).

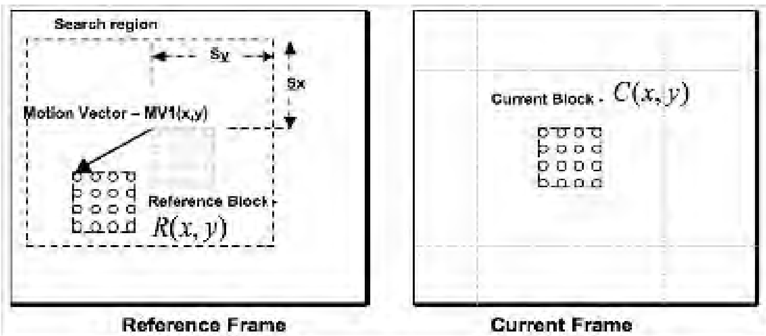

Fig 1. Block Diagram of BMA

The necessity for block matching technique is rising. FS algorithms using SAD block can be performed either by the usage of 1-D or 2-D systolic or systolic-like designs. The 1-D design provides less complex data arrangement and simpler structures i.e., reduction in gate-counts. This 1-D design is used for its lesser silicon area and the chip-size in the portable devices.

III. SAD IMPLEMENTATION

In this paper a SAD architecture has been designed and constructed. The 1-D SAD design contains sixteen Absolute Difference (AD) block. An important feature of 1-D SAD design is it merges with Accumulators followed by some additional registers including a small amount of hardware in the data path of a single AD block, to store the SAD values and finally these values are compared using a comparator to find the best result i.e., MV. The organization of the current data and reference data is similar to a traditional 1-D design. The SAD operation is split into two stages namely, Absolute stage and Sum stages.

Absolute Stage:

In this stage, the current data and the reference data are considered to be unsigned 8-bit data. In this paper a carry generator is used to calculate the carry, based on the carry-propagate algorithm. Based on the outcome of the carry generator, the right value (A or B) is bit inverted and added to the remaining value (B or A). This diagrammatic representation of Absolute stage is illustrated in Fig 2.

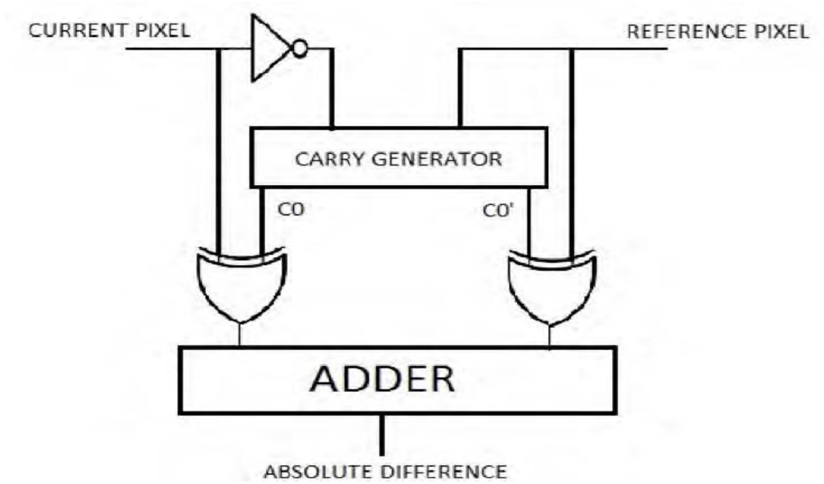

Fig 2. Block diagram of SAD

Sum Stage:

In this stage, all the summation terms resulted by the absolute stage is summed up. In this paper, we use carry-look ahead adder for the sum stage and pipelining method is implemented. Absolute stage output is stored in a register and accumulator is used to accumulate all the values with some addition of registers and the outputs are given to the comparators to find the minimum SAD value. Utilizing this technique to the MB shown in Fig 3, outcome is the single MV for a single AD block with Accumulator, all with 4×4 block sizes. The accumulated values are then compared to calculate MV values for additional block sizes. Thus the designed 1-D SAD architecture has an increase in speed and reduction of hardware.

The single AD block calculates the difference between the Current pixel value and Reference pixel value on all clock cycles. The current MB data and search area data are expressed by C(x,y), and R(x,y). The pixel standards of the current MB data are represented from p0 to p255. For example, the computation of the SAD value for b0 contains pixels p0 - p3, p16 - p19, p32 - p35 and p48 - 51. The diagram shows similar for block b1, it contains pixels p4 - p7, p20 - p23, p36 - p39 and p52 - p55, and so on (refer Fig 3).

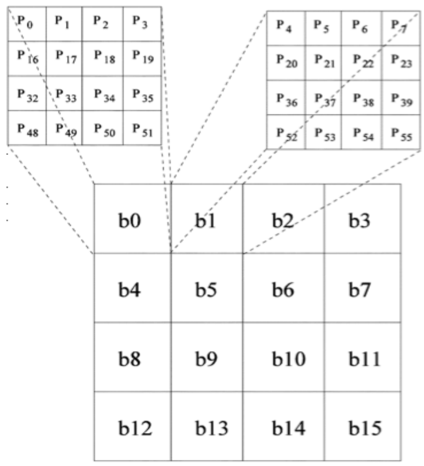

Fig 3. Pixel values in the blocks (namely, p0 and p1)

The SAD uses a three-step method. The initial stage of the AD is to calculate the difference between the current pixel data and reference pixel data. These outcomes are then given as the input to second step where they are accumulated and saved in the registers. The purpose of register and Accumulators is to confirm that as soon as the computations has been completed these are saved and served back to calculate the AD for all the sub-blocks viz., b0 - b15. Then all the outputs of the accumulated values are compared using comparators to find the minimum SAD value. The design has reduction in hardware and also there is an increase in speed when compared with the previous works. The results obtained are checked using ModelSim Simulator, for its functionality. The SAD process delivers good coding results and the rise in computation is also high.

IV. Block Diagram and Results of SAD Architecture

The architecture is designed for ME using SAD in this paper is shown below in Fig 4.

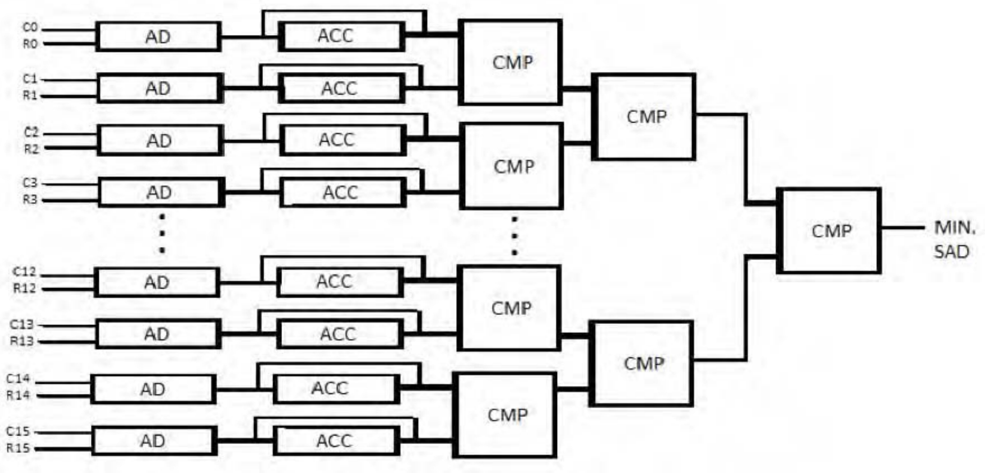

Fig 4. Block Diagram of Pipelined SAD Architecture

Based on the design flow and Verilog HDL code, the designed architecture can be functionally verified by using ModelSim Simulator. The TABLE I show the SAD architecture performance as follows.

TABLE I
SAD ARCHITECTURE PERFORMANCE

TECHNOLOGY	TSMC 45 nm CMOS
ALGORITHM	FIXED BLOCK SIZE MOTION ESTIMATION using SAD ARCHITECTURE
MAX. FREQUENCY (MHz)	176.05 MHz
GATE COUNT	3k
AREA OBTAINED	10321.536
LEAKAGE POWER	270.743 nW
DYNAMIC POWER	1092296.453 nW
TOTAL POWER	1092567.196 nW

A comparison between this design and previous SAD designs are presented in Table II. This paper with 1-D architecture has an increase in speed as compared to [2] and [3] with the highest working frequency of 176.05 MHz and also has a gate count of around 3k. This shows that the designed hardware has increase in speed and also there is a reduction in number of gates.

TABLE II

	[2]	**[3]**	**PROPOSED**
Process	Altera FPGA	Xilinx Virtex II FPGA	TSMC 45 nm CMOS
Frequency (MHz)	120 MHz	68 MHz	176.05 MHz
# Of Gate Count	7k	5.8k	3k

V. Conclusion

In this paper, we designed a high performance and low cost hardware architecture for the real-time implementation of an SAD module based hierarchical ME algorithm for H.264 / MPEG4 Part-10 video coding. This hardware is designed to be as a part of a complete H.264 video coding system for portable applications like televisions, Internet, broadcasting, etc. The designed architecture is implemented in Verilog Hardware

Description Language. The Verilog RTL code is verified by using ModeSim Simulator. This paper is implemented using TSMC 45nm CMOS technology using ASIC flow in CADENCE RTL compiler tool and the obtained working frequency is about 176.05 MHz and the gate count achieved is around 3k. And this implementation can process a real time (30 fps) H.264 ME on 1920 × 1080 progressive HD videos.

REFERENCES

[1] Coding of Moving Pictures and Audio, ISO/IEC Std. 14 496-10, 2006.

[2] Cao Wei; Mao Zhi Gang, "A novel SAD computing hardware architecture for variable-size block motion estimation and its implementation with FPGA," *ASIC, 2003. Proceedings. 5th International Conference on*, vol.2, no., pp.950-953 Vol.2, 21-24 Oct. 2003.

[3] Sinan Yalcin; Ates, H.F.; Hamzaoglu, I., "A high performance hardware architecture for an SAD reuse based hierarchical motion estimation algorithm for H.264 video coding," *Field Programmable Logic and Applications, 2005. International Conference on*, vol., no., pp.509-514, 24-26 Aug. 2005.

[4] THE H.264 ADVANCED VIDEO COMPRESSION STANDARD, Second Edition-Iain E. Richardson, Vcodex Limited, UK.

[5] Wong, S.; Stougie, B.; Cotofana, S., "Alternatives in FPGA-based SAD implementations," *Field-Programmable Technology, 2002. (FPT). Proceedings. 2002 IEEE International Conference on*, vol., no., pp.449-452, 16-18 Dec. 2002.