p-ISSN : 2335-1357 e-ISSN : 2437-1122

Mediterranean Journal of Modeling and Simulation

Med. J. Model. Simul. 11 (2019) 009-019

Efficient Absolute Difference Circuit for SAD computation on FPGA

Jaya Koshta [a] *, Kavita Khare [a]

[a] Maulana Azad National Institute of Technology,Nehru Nagar, Bhopal,462003,India

ARTICLE INFO

Article history :
Received April 2019
Accepted September 2019

Keywords :
HEVC ;
Motion estimation ;
Sum of absolute difference ;
Sarallel prefix adders ;
Brent Kung Adder.

ABSTRACT

Video Compression is very essential to meet the technological demands such as low power, less memory and fast transfer rate for different range of devices and for various multimedia applications. Video compression is primarily achieved by Motion Estimation (ME) process in any video encoder which contributes to significant compression gain. Sum of Absolute Difference (SAD) is used as distortion metric in ME process. In this paper, efficient Absolute Difference(AD) circuit is proposed which uses Brent Kung Adder(BKA) and a comparator based on modified 1's complement principle and conditional sum adder scheme. Results shows that proposed architecture reduces delay by 15% and number of slice LUTs by 42 % as compared to conventional architecture. Simulation and synthesis are done on Xilinx ISE 14.2 using Virtex 7 FPGA.

1. Introduction

To meet the technological demands such as low power, less memory and fast transfer rate for a wide range of applications including the growing demand of High Definition(HD)(1080p) to Ultra-High Definition(UHD)(4K and 8K), resulted in creation of stronger needs for better video compression efficiency. HEVC or H.265is a video compression standard designed to substantially improve coding efficiency when compared to its precedent, the Advanced Video Coding (AVC) or H.264.HEVC is a block-based video compression designed to support higher resolutions and it can achieve 50% bit rate saving compared to H.264/MPEG-4 AVC for the same video quality [1][2]. This considerable increase in performance is due to the many enhanced techniques and methodologies that have been introduced in H.265/HEVC. Some of these enhancements are included in the motion estimation process, which is one the most complex and time consuming block in video encoding. The objective of the ME unit is to find the best matched block in the reference (past/future) frame search window (region of interest), for every block of the current frame such that there constructed frame contributes to the lowest residual

*Email : jayakoshta15@gmail.com ; Tel. : 091-9406542019

information [3].The bottleneck for the ME system design lies in the implementation of an appropriate SAD architecture. For block based ME, the Sum of Absolute Difference(SAD) is the generally used metric which adds up the absolute differences between corresponding elements in a candidate and reference block in video frames. However with the increase in the coding block size to 64x64 in HEVC, compared to 16x16 in H.264/AVC, results in greater complexity in ME process. Thus in HEVC for ME process, the number of SAD computations vastly increases resulting in increase in the processing delay, power consumption and hardware complexity. Thus, a pragmatic strategy to elevate the performance of H.265/HEVC relies on enhancing the performance of the core component of the motion estimation engine which is the block matching unit.

Different VLSI architectures for SAD computations are available in the literature where there is a lot of trade-offs between the speed, power and area during the hardware implementation. Due to the increasing demand for the portable devices with low power consumption and high performance, the circuits are optimized according to the applications. In this paper, implementation of absolute difference circuit on FPGA is proposed to increase speed performance and to minimize the amount of occupied resources on FPGA for SAD calculation.

2. Related Work

SAD is one of the most popular techniques used for motion estimation in digital video encoding systems. There are large number of methods available for SAD computation where there are tradeoffs between speed, power and area during the hardware implementation. Typically, the SAD computation consists of first computing the absolute difference between corresponding pixels in current and reference video frame.

The calculation of SAD from the current and reference block is performed using equation (1),

$$SAD = \sum_{i=1}^{M} \sum_{j=1}^{N} |CB(i,j) - RB(i,j)| \tag{1}$$

Where,
CB - Current Block,
RB –Reference Block,
$N\ X\ M$ - block size of current and reference block,
i, j - two dimensional coordinates of block.

Generally, the SAD operation basically consists of first computing the absolute difference $|CB(i,j) - RB(i,j)|$ and then summing up these in a multi-input addition.

For absolute difference calculation, one method is to detect the smaller operand in the absolute difference computation $|CB - RB|$ and to subtract it from the larger operand [4], [5]. The other method comprises of complimenting the smaller of the two numbers and then performing addition of two numbers followed byplus one to compute the absolute difference[6]. To compute the absolute difference for SAD, in [7] a novel architecture is optimized for realizing efficient absolute difference circuits in Virtex-5 FPGA devices which uses the 6-input look-up tables available within the chosen devices family to maximize

speed performance and to minimize the amount of occupied resources. In [8] an improved architecture for efficiently computing the sum of absolute differences (SAD) on FPGAs is proposed based on a configurable adder/subtractor implementation in which each adder input can be negated at runtime . The SAD architecture proposed in this paper provides a significant resource reduction on current FPGAs. In [9] FPGA design for fast computing of the minimum SAD is proposed. The hardware unit proposed is intended to augment a general-purpose core and with the use online arithmetic (OLA) it is possible to implement a full 16 X 16 macroblock SAD in a single FPGA device and it permits to speed up the computation by early truncation of the SAD calculation when the involved candidate is bigger than the current reference SAD. In [10] pipelined SAD architecture for efficient SAD calculations is proposed where consecutive pipeline stages perform the addition of absolute differences to obtain SAD of 8 X 8 block.

Table 1 summarises the above metioned related work on SAD computation. Review work motivates for a proposal of high speed compact AD circuit on FPGA for SAD calculation. In this paper, AD circuit on FPGA is proposed to increase speed performance and to minimize the amount of occupied resources on FPGA for SAD calculation.

Table 1 – Related work on SAD computation

Ref.	Architecture	Advantages	Disadvantages
[7]	6-input look-up tables on Virtex 5	High speed and compact	Optimized for only Virtex 5
[8]	Configurable adder/ subtractor on Virtex 6	Resource reduction on current FPGAs	SAD 1X2 implemented
[9]	Online arithmetic for 16 X 16 SAD on Virtex 2	High speed	Large Area
[10]	Pipelined SAD architecture	High speed and low power	Large area

3. Proposed Architecture

Hardware architecture for computing the absolute difference between corresponding pixels in current and reference video block is proposed. The method used for AD calculation is as used in [6] where adder and comparator form the basic component as shown in Figure 1. The 8-bit comparator compares two numbers and returns the 1's complement of the smaller number and the larger number as it is.

Proposed Architecture : In this architecture :

– Ripple Carry Adders in AD circuit (Figure 1) are replaced by Brent Kung Adder (Type of Parallel Prefix Adders-PPA).

– 8-bit conventional comparator in Figure 1 is replaced by comparator based on modified 1's complement principle and conditional sum adder scheme.

These replacements result in reduction in delay and reduces

LUTs count used in the circuit.

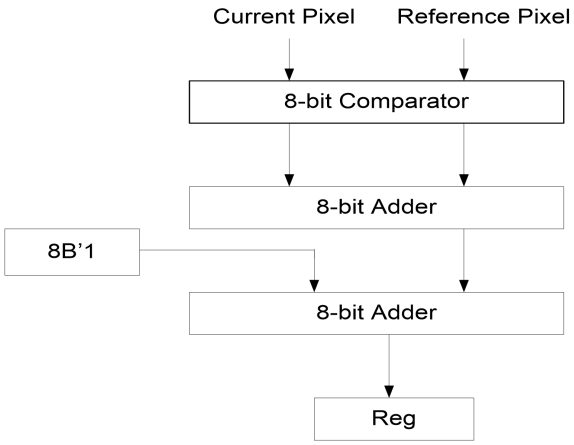


Fig. 1 – AD circuit block diagram

PPA :

PPA (Parallel Prefix Adder) circuits use a tree network to reduce the latency to O(log2n) where 'n' represents the number of bits. PPA employs 3-structural stages as shown in Figure 2. The first stage at the top is used for computing generate and propagate signals as given in equation (2) and (3) exactly as in Carry Lookahead Adder (CLA) where a and b represents binary digits.

$$g = a.b \tag{2}$$

$$p = a \oplus b \tag{3}$$

The carry bits are calculated in the second stage. In this stage, to formulate the operation of the prefix adders, "prefix operator" which is represented as "." is used. The "prefix operator" function has two essential properties to keep the computational operation faster. The first one is called the associative property and the second one is idempotency property. Using the advantages of these properties, carry-out can be found at a depth proportional to log2(n) [9]. The final summation is obtained in the last stage. The associative and idempotency properties of "." operator allow carry output to be computed in a different number of levels or simply depth. Therefore, various topologies of prefix adders can be designed which are mentioned in the literature and are inspiring to VLSI designers because of their minimum depth and delay. The logical structures used in prefix adders scheme consists of black cell, gray cell and white cell as defined in Figure 3 [10].

The logical structure of black cell can propagate and generate signals while the logical structure of gray cell can only generate signals. The white cell (buffer)is used for loading the signal out. In parallel prefix scheme, generate and propagate signal scan be grouped in multiple ways to get the same correct carry signals. Based on different methodologies of grouping these signals, different prefix architectures can be created. The parallel form of obtaining the carry bit makes PPAs perform addition arithmetic faster. There are many types of PPA such as Brent Kung, Kogge Stone, Ladner Fisher, Hans Carlson and Knowles [10].

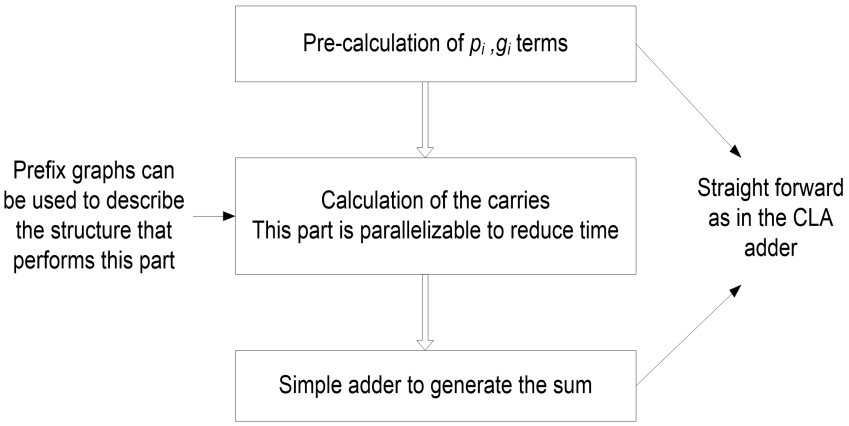


Fig. 2 – Parallel Prefix Adder stages

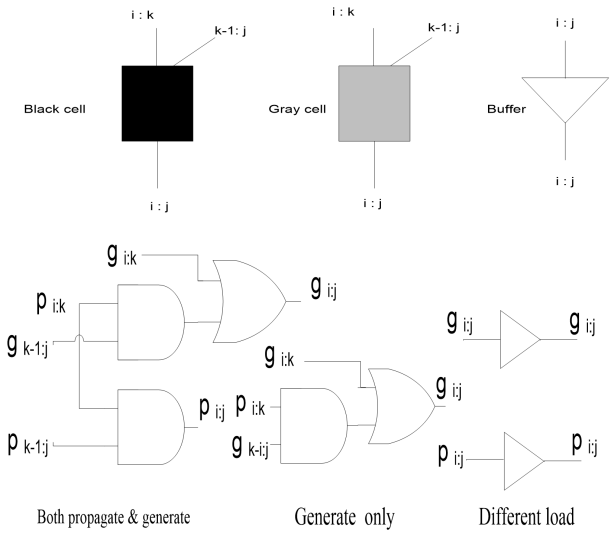


Fig. 3 – Cell definitions for parallel-prefix scheme [10]

Out of many types of PPA, Brent Kung Adder(BKA) is used as for 8-bit implementation it has less delay compared to other types of PPA [11][12].

In BKA propagate signals and generate signals are combined into groups of two by using the associative property. The first stage in BKA includes computation of generate and propagate signals corresponding to each pair of bits in a and b as given by equation (2) and (3). The second stage ie. prefix carry tree includes computation of carries corresponding to each bit. Equations (4) and (5) below shows how propagate and generate signals are calculated in BKA,

$$p_{i:j} = p_{i:k}.p_{k-1:j} \tag{4}$$

$$g_{i:j} = g_{i:k} + (p_{i:k}.g_{k-1:j}) \tag{5}$$

These propagate and generate signals given in equation (4) and (5) are calculated using black cell, gray cell and white cell as defined in Figure 3. The last stage ie. post processing

stage includes computation of sum bits which is given by the equation (6).

$$s_i = p_i \oplus c_i \tag{6}$$

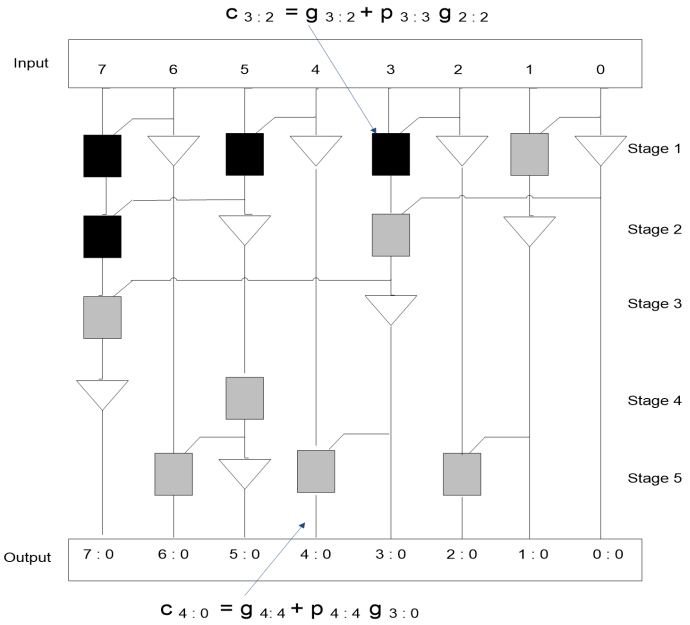
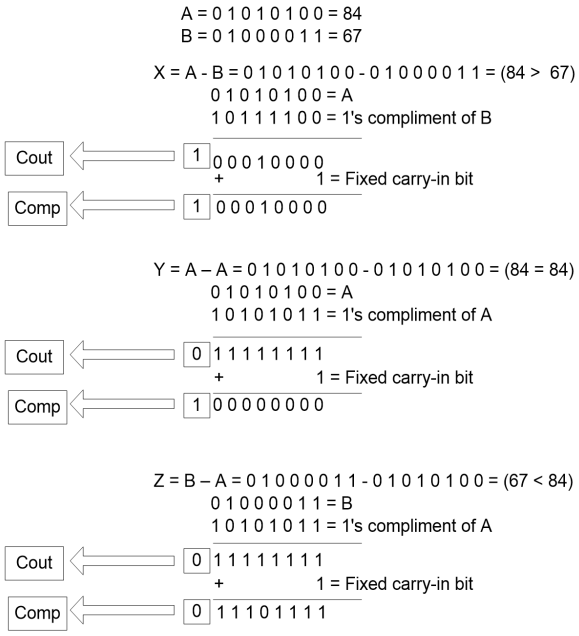


Fig. 4 – 8-bit Brent-Kung adder(BKA) tree [13]

Figure 4 shows 8-bit BKA tree adder which has lower delay as compared to Ripple Carry Adder(RCA). Also, apart from using BKA, a new efficient comparator based on modified 1's complement principle and conditional sum adder scheme is used in this architecture as discussed below.

The proposed architecture apart from using BKA,also uses comparator (based on modified 1's complement principle and conditional sum adder scheme). In this modified 1's complement scheme, if $A > B$, bit $Cout = 1$ and if $A \leq B$ bit $Cout = 0$ so the only concern is about carry out bit information as shown in Figure 5. This method always adds a fixed carry after modification, so if $A \geq B$, bit $Comp = 1$ and if $A < B$, bit $Comp = 0$ [14]. For realization of this comparator conditional sum adder scheme has been used that provides a logarithmic increase in speed for addition[15].

The principle behind this scheme is to generate two sets of outputs for a given group of k bits operands. Each set includes k sum bits and an outgoing carry. The one set assumes that the eventual in coming carry will be zero, while the other assumes that it will be one. Depending upon the incoming carry correct set of outputs (out of the two sets) is selected without waiting for the carry to further propagate through the k positions as shown in Fig.6.



Status bit	Cout	Comp
A>B	1	1
A=B	0	1
A<B	0	0

Fig. 5 – Modified l's complement method for improved comparator design

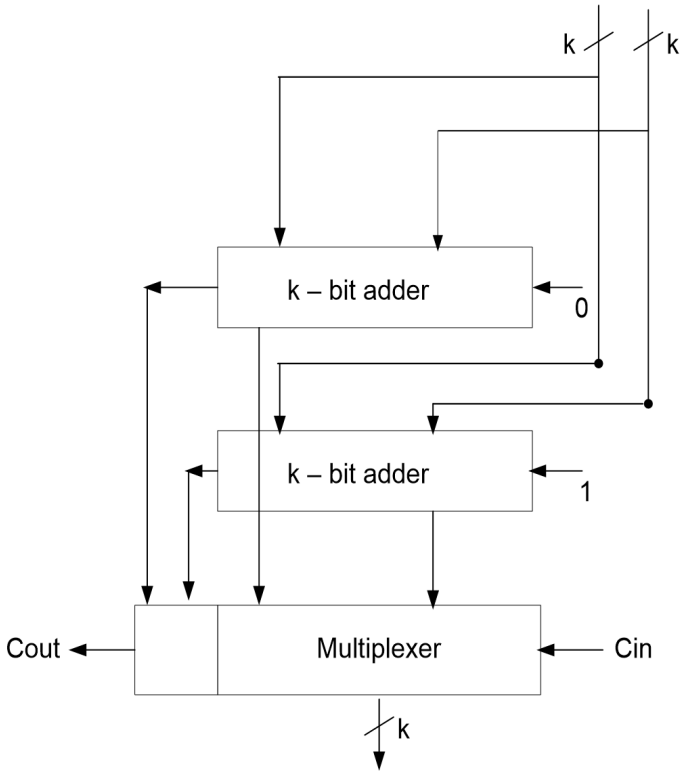


Fig. 6 – Conditional sum adder scheme

However, this scheme is generally not applied to long n-bits operand at the beginning of the add operation, since it will add delay as the carry propagates through all n positions before making the selection for correct output. Generally, the given n bits are divided into smaller groups to apply this conditional adder scheme separately. Thus, the serial carry-propagation inside the separate groups can be done in parallel, reducing the overall execution time. The outputs of the subgroups are then combined to generate the output of the final output.

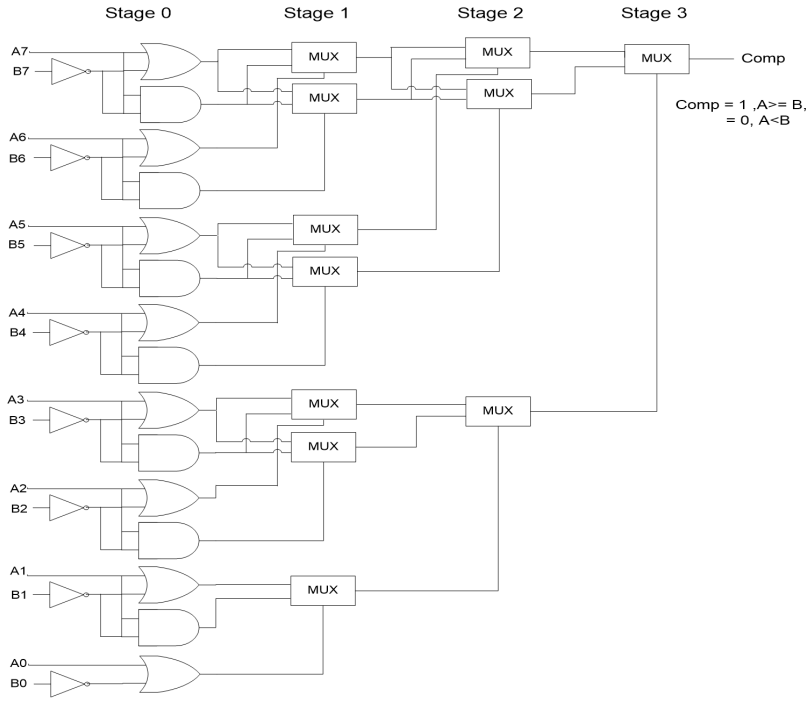


Fig. 7 – Modified comparator architecture

Figure 7 shows comparator architecture using modified 1's complement and conditional sum adder design. The 8-bit comparator needs 11(=1+3+7) 2-to-1 multiplexers and eight inverters to generate complementary values of input B. Originally Carry = AB + AC + BC = AB + (A + B) C and if C= 0 then Carry =AB or if C=l then Carry = AB + (A+B) =A + B. The sum of MUX gates of N-bit comparator is,

$$\sum_{k=1}^{M} \left(2^k - 1\right) where\ M = \ log_2 N \tag{7}$$

This architecture reduces delay and also provides significant reduction in resource utilization in FPGA.

4. Simulation and Results

Simulations of the conventional and proposed architecture for AD circuit implementation have been carried outusing Verilog HDL programming in Xilinx ISE 14.2 platform and implemented on Virtex7 FPGA. Adequate testing of each design was done to verify

correct operation. Figure 8 shows the simulation result of proposed AD circuit.

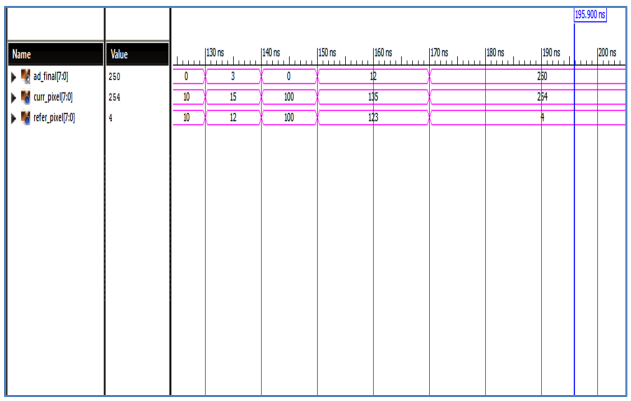


Fig. 8 – Simulation result of Absolute Difference Circuit

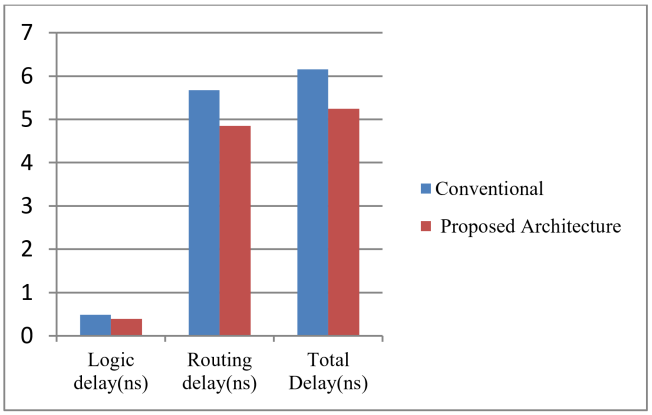


Fig. 9 – Graphical representation of delays in proposed architecture with conventional architecture

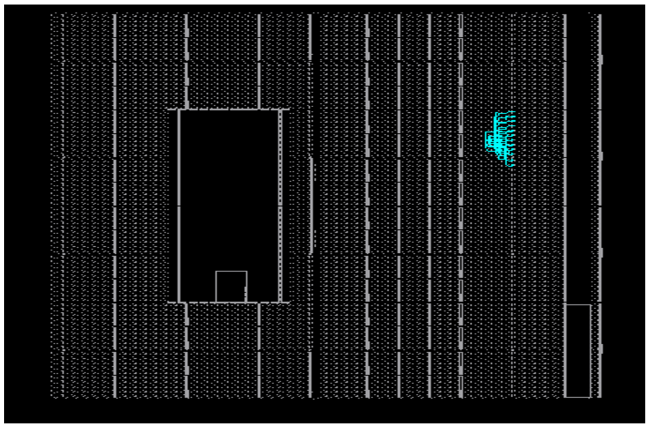


Fig. 10 – Implementation of proposed AD architecture on FPGA

Table 2 – Comparison of proposed AD architecture architecture with conventional architecture

Architecture	**Logic**	**Routing**	**Total**	**Number of**
	delay(ns)	delay(ns)	delay(ns)	slice LUTs
Conventional	0.484 (8.0% logic)	5.673 (92.0%route)	6.157	50
Proposed Archiecture	0.391 (7.5% logic)	4.849 (92.5% route)	5.240	29

5. Conclusion

VLSI architecture for SAD computations is proposed in this paper for reducing delay and area and is implemented on Virtex 7 FPGA. The proposed architecture reduces delay and provides significant reduction in resource utilization in FPGA. Synthesis results shows that proposed architecture reduces delay by 15% and number of slice LUTs by 42 % as compared to conventional architecture.

REFERENCES

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T.Wiegand,"Overview of the high efficiency video coding (HEVC)standard," IEEE Trans. Circuits Syst. Video Technol., vol.22, no. 12, pp. 1649-1668, December 2012.

[2] I. Richardson, "HEVC : An introduction to high efficiency video coding," 2001, https ://www.vcodex.com/h265.html

[3] N. Purnachand, L. N. Alves, and A. Navarro, "Fast Motion Estimation Algorithm for HEVC", IEEE International Conference on Consumer Electronics-Berlin (ICCE-Berlin), September 2012.

[4] S.Wong, B. Stougie, and S. Cotofana, "Alternatives in FPGA-based SAD Implementations," in IEEE International Conference on Field-Programmable Technology (FPT),IEEE, 2002, pp. 449–452.

[5] S. Vassiliadis, E. A. Hakkennes, J. S. S. M. Wong, and G.G.Pechanek, "The Sum-Absolute-Difference Motion Estimation Accelerator," in Euromicro Conference, 1998. Proceedings, 24th. IEEE, 1998, pp. 559–566.

[6] Ahmed Medhat, Ahmed Shalaby, Mohammed S. Sayed, Maha Elsabrouty and Farhad Mehdipour. "A Highly Parallel SAD Architecture for Motion Estimation in HEVC Encoder", Circuits and Systems (APCCAS), IEEE Asia Pacific Conference, 280 - 283, 2014.

[7] Stefania Perri , Paolo Zicari , Pasquale Corsonello, "Efficient Absolute Difference Circuits in Virtex-5 FPGAs", IEEE, 2010.

[8] Martin Kumm, Marco Kleinlein and Peter Zipf,"Efficient Sum of Absolute Difference Computation on FPGAs",26th International Conference on Field Programmable Logic and Applications.

[9] Joaquin Olivares, Ignacio Benavides and et.al., "Minimum Sum of Absolute Differences implementation in a single FPGA device", Dept. of Electro-technics and Electronics, University of Cordoba, Spain.

[10] P.Jayakrishnan and Harish. M. Kittur,"Pipelined Arch. for Motion Estimation in HEVC VideoCoding", Indian Journal of Science and Tech,August 2016.
[11] Geeta Rani and Sachin Kumar,"Delay analysis of parallel-prefix adders", International Journal of Science and Research (IJSR), 3(6) :2339-2342, 2014.
[12] Nurdiani Zamhari, Peter Voon, Kuryati Kipli, Kho Lee Chin, Maimun Huja Husin,"Comparison of Parallel Prefix Adder (PPA)", Proceedings of the World Congress on Engineering 2012,Vol II.
[13] R. P Brent & H. T. Kung, "A Regular Layout for Parallel Adders", IEEE Trans. Computers, Vol. C-31, pp 260-264, 1982.
[14] Shun-Wen Cheng, "A High-Speed Magnitude Comparator with Small Transistor Count", in Proceedings of IEEE international conference ICECS, 1168 - 1171 Vol.3, Dec 2003.
[15] J.Sklansky, "Conditional-Sum Addition Logic," IRE Transactions on Electronic Computers, Vol. EC-9, No. 2, pp. 226-231, June,1960
[16] S. Rehman ; R. Young ;C. Chatwin ;P. Birch, "An FPGA Based Generic Framework for High Speed Sum of Absolute Difference Implementation", Europ. Jour Scient. Res., vol.33, no.1, 2009.
[17] Manjunatha, D. V., and G. Sainarayanan. "Low-Power Sum of Absolute Difference Architecture for Video Coding", Emerging Research in Electronics, Computer Science and Technology. Springer India, 2014. 335-341.
[18] LiYufei, Feng Xiubo and Wang Qin,"A High-Performance Low Cost SAD Architecture for Video Coding", IEEE Transactions on Consumer Electronics, pp. 535-541, Vol. 53, No. 2, May 2007.
[19] Jarno, Vanne,Eero Aho, Timo D. Hamalainen and Kimmo Kuusilinna, "A High-Performance Sum of Absolute Difference Implementation Motion Estimation", IEEE Transactions on Circuits and Systems for Video Technology, pp. 876-883, Vol. 16, No. 7, 2006.
[20] Elhamzi W., Dubois J., Miteran J "An efficient low- cost FPGA implementation of a configurable motion estiation for H.264 video coding", Springer Journal of Real-Time Processing,Vol :9, No :1, pp. 19–30,2014.
[21] Moorthy T., Ye A, "A scalable architecture for variable block size motion estimation on field-programmable gate arrays" , IEEE Canadian Conference of Electrical and Computer Engineering (CCECE),Niagara Falls, May, pp.1303–1308,2008.
[22] Davis P., Sangeetha M. ,"Implementation of Motion Estimation Algorithm for H.265/HEVC",International Journal of Advanced Research in Electrical,Electronics and Instrumentation Engineering. Vol :3,No :3, pp.122–126,2014.