



## VLSI Implementation of Motion Estimation Using Level Convertor with CSLA Adder

Dipti Yashodhan Sakhare<sup>1\*</sup>

<sup>1</sup>*MIT Academy of Engineering, Alandi (D), Department of Electronics Engineering,  
School of Electrical Engineering, Pune, Maharashtra, India*

\* Corresponding author's Email: [dysakhare@etx.maepune.ac.in](mailto:dysakhare@etx.maepune.ac.in)

---

**Abstract:** Nowadays, Motion Estimation (ME) plays a vital role in traffic signals, movie making, and security purpose to identify the motion of an object. In this paper, Smart and Low power (SLP) based ME is introduced with Level Convertor and Carry Select Adder (LC-CSLA-ME). Normal SLP architecture requires more power for performing ME, but in the LC-CSLA-ME architecture, 0.9v supply voltage is used to carry-out the ME. The LC is used to supply the required voltage to the respective sub blocks, which helps to reduce the power consumption of the overall architecture. Instead of using normal adder, CSLA adder is used in prediction and classification unit for improving the hardware utilization. The performance of Application Specified Integrated Chips (ASIC) and Field Programmable Gate Array (FPGA) analysed for both existing and proposed methods. In 180nm technology, LC-CSLA-ME architecture occupied 0.065mm<sup>2</sup> area, 2.10mW power, and 104ms delay. The different dataset such as Johnny, Vidyo 1, and Vidyo 4 is analysed for ME. The Coding Time Saving (CTS) evaluated for both existing and LC-CSLA-ME architectures. In the LC-CSLA-ME, average CTS of Operating Power Points (OPP1, OPP2, OPP3, OPP4, and OPP5) are 34.021, 54.316, 67.021, 75.027, and 80.071 that is high compared to the existing methods.

**Keywords:** Application specified integrated chips, Carry select adder, Field programmable gate array, Level convertor, Motion estimation.

---

### 1. Introduction

In recent years, Motion Estimation (ME) is one of the important technology for vehicle tracking, traffic, mobile robots, and film industry [1]. In the film industry, High Definition (HD) video becomes more important to attract the audience. So, the ME process also require to give efficient editing [2]. ME includes the Block matching algorithm which is used to exploit the redundancy, but it requires more computation time [3]. The spatial and temporal redundancy between the frames removed in ME [4]. The Fractal ME algorithm provides the ME sub pixel accuracy of smart camera that used in accelerators. This algorithm improves the computation intensive [5-7]. With the help of fast search algorithm, most of the ME has been performed. The Binary search technique used for ME that is represented by one or two bits which helps to reduce

the computational complexity [8]. Optimal flow multichannel has been designed for improving the timing of ME [9].

Most of the existing algorithms used for ME are Bilateral ME [10], ego ME [11], variable block size ME [12], optical flow estimation [13], full search ME [14], and particle ME [15]. But, these kind of methods doesn't have any motion pattern that affects the quality of the video. These methods often affect the shadow noise which degrade the frame level performances. In VLSI architecture, the design of ME requires more hardware and power. To overcome this problem, Level Convertor with Carry Select Adder based motion estimation (LC-CSLA-ME) is introduced in this paper. Level converter is most important design to maintain the entire architecture with less amount of power. The separation of voltage also performed in LC. Moreover, the hardware utilization also need to improve in proposed method.

Because of that, optimal CSLA adder is used which helps the architecture working effectively with less area. The ASIC and FPGA performances improved in LC-CSLA-ME compared to conventional methods.

The rest of the paper is organized as follows: Section 2 elaborates the literature survey, section 3 explained the problem statement, section 4 describes the proposed method, section 5 discusses about the

experimental results, and Conclusion is summarized in section 6.

## 2. Literature review

Researchers have suggested several methods on the ME architecture. This section presents a brief evaluation of some significant contribution in the field of ME architecture.

Author	Methodology	Advantage	Disadvantage
Rohan Mukherjee et al. [16]	root pattern search algorithm for motion estimation	Adopted Root Pattern Search algorithm (ARPS) was introduced to reduce the time complexity. This work was tackle the adaptive nature algorithms which can be used to detect the motion form high quality videos.	This works required more memory to store the intermediate data information.
P.P Shiju et al. [17]	fixed mesh based motion estimation algorithm for video encoding process	This mesh based models are used in non-translation motion estimation process. Synopsis tool was used for ASIC implementation which consume 344.8 MHz frequency and 4.89mW power.	This method provides more shadow noise which affects the video quality.
Baishik Biwas et al. [18]	efficient adaptive rood pattern search algorithm for motion estimation	This work is applied for 352x288 CIF frames which can extended for high resolution. This architecture was used 165 slices registers, 320 MHz frequency, 273 slice LUT and it can process 240 frames per second	This architecture need more supply to operate and evaluate the performances.
Shiju padmanabhan et al. [19]	Motion estimation using block matching algorithm.	This architecture was designed both ASIC and FPGA implementation. ASIC result was verified in 180nm technology and FPGA result was verified in Virtex 4 devices	This architecture occupies more area and more power.
Grzegorz pastuszak and Maciej trochimiuk [20]	Motion estimation for H.265 4K-UHD decoder.	This design allows the processing of 2160p@30fps videos at the clock frequency of 400 MHz. The motion estimation blocks are consider as 8x8 blocks which helpful for getting high throughput	The accuracy of the proposed method is too low

## 3. Problem statement

This section describes about the issues of ME and also detailed about how the proposed methodology gives solution to the described problems. The concerns of ME architectures are detailed below.

- Complex architecture doesn't perform ME effectively. The block information is not explained properly.
- More power supply is required for all the sub block architectures.
- Normal digital adder requires more area.
- A number of RAM is required to store the data and perform the further process of ME operation.
- Final output has some degradation and information loss.

**Solutions:** To overcome the above mentioned drawbacks, LC-CSLA-ME architecture is implemented in this work. LC is used to reduce the power consumption of the entire work. The optimal CSLA adder is used to perform the Incrementer operation. Finally, the process speed and area is improved through this method.

## 4. LC-CSLA-ME methodology

This research paper introduced the design of ML-based VLSI architecture of high-performance and power-efficient (HPPE) Smart ME Controller (SMC). The principle of the proposed SMC engineering is to limit the coding data transmission usage and expand

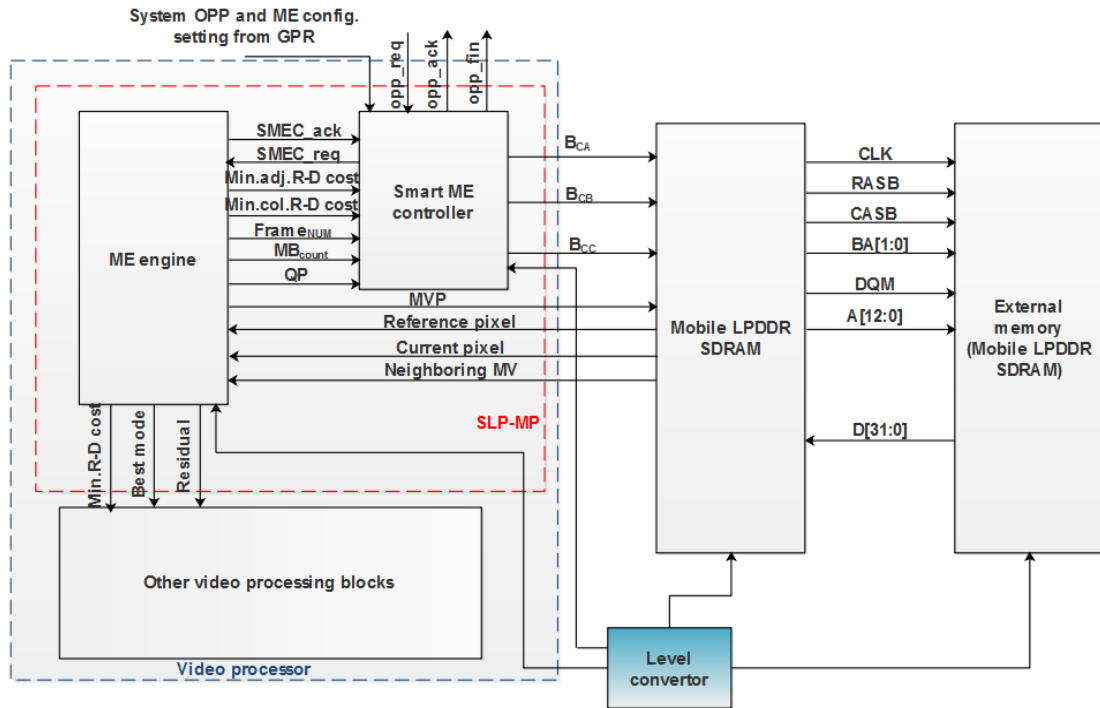


Figure. 1 Block diagram of LC-CSLA-ME methodology

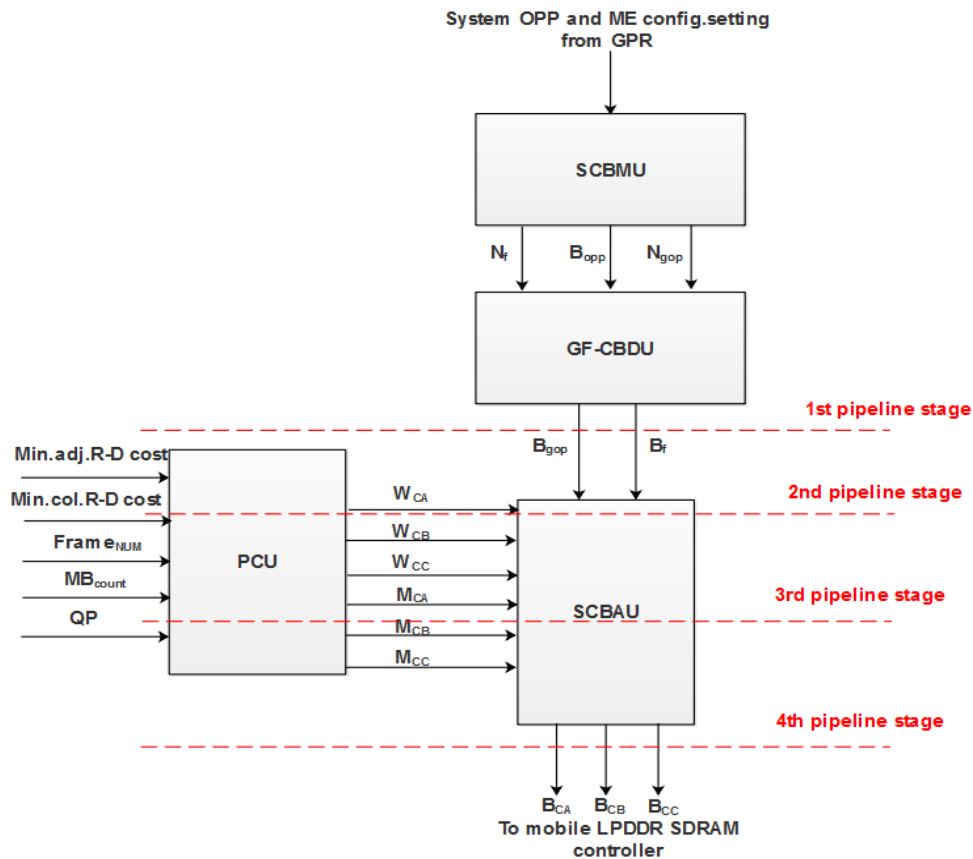


Figure. 2 Block diagram of SMC

the coding execution under on-demand bandwidth problems using Power Management (PM) technology in an intelligent portable SoC. The Block diagram of LC-CSLA-ME methodology is shown in Fig.1.

The configuration comprises of two building blocks: the ME engine and the SMC. It is important that the proposed SLP ME configuration is effectively coordinated into a video SoC and versatile

AP by wrapping the two building blocks such as Low Power Double Data Rate (LPDDR) SDRAM controller and LPDDR SDRAM. Initially, the input frames are read in MATLAB which is converted into the binary value. This binary value is stored in external memory which is shown in Fig. 1. That 32-bit binary value is given to the mobile LPDDR SDRAM which outputs (Current pixel, reference pixel, and neighbouring MV) are performed ME engine. The remaining process is performed based on Fig. 1.

#### 4.1 SMC

In this paper, five predefined OPPs used by the SMC to achieve low-power design objective in different PM standards. The block diagram of SMC is shown in Fig. 2. The output of the ME is given to the SMC which contains, Smart Coding Bandwidth Monitoring Unit (SCMBU), GOP- and Frame Layer Coding Bandwidth Determination Unit (GF-CBDU), Prediction and Classification Unit (PCU), and Smart Coding BA unit (SCBAU). This SMC is worked based on Fig. 2 operation. The individual block diagram of PCU is given in below section.

##### 4.1.1. Smart coding bandwidth monitoring unit

The SCBM unit is used to monitor the present power situation and it sets the accurate OPP setup for more ME activities. The available coding data transmission can be obtained from predefined OPPs which are stored in General Purpose Registers (GPRs) or on-chip ROM within an intelligent versatile SoC. The devised SCBMU design receives the power situation change signals from the PMU along with the ME arrangement settings inside the AP. At this point, the SCBMU sends the comparing coding data of  $B_{opp}$ ,  $N_f$ , and  $N_{gop}$  to the GOP- and frame layer coding for determining the bandwidth.

##### 4.1.2. GOP- and frame layer coding bandwidth determination unit

This is used to allocate the coding bandwidth for Group of Picture (GOP) and frame layer. Bandwidth Allegation (BA) is more important to design low power VLSI architecture. Here, BA for GOP and frames are distributed uniformly. The uniform distribution problem is shown in Eqs. (1) and (2).

$$B_{gop} = \frac{B_{opp}}{N_{gop}} \tag{1}$$

$$B_f = \frac{B_{gop}}{N_f} \tag{2}$$

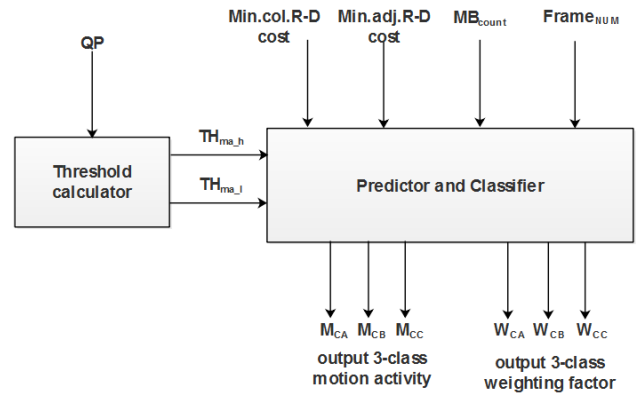


Figure. 3 Block diagram of PCU

Where,  $B_{gop}$  is represented as GOP of BA, and  $B_f$  is frame layer of BA.

$B_{opp}$  = Predefined available coding bandwidth

$N_{gop}$  = Number of GOP

$N_f$  = Frames within on OPP and one GOP

The GF-CBDU is used to allocate the objective coding data transfer capacity for GOP ( $B_{gop}$ ) and frame ( $B_f$ ) coding and it characterized in Eqs. (1) and (2) for the whole ME task inside one OPP situation. Every situation contains various GOPs, and every GOP incorporates various frames. The reduction of hardware utilization is achieved by fixing quantities of GOPs into multiple of two. Henceforth, we set  $N_{gop}$  and  $N_f$  to the most often utilized settings of  $2^{gop}$  and  $2^f$  for VLSI execution with a few shifters. In this way, the grouped shifters are utilized for various video applications and various frame rates. Thus, the GF-CBDU configuration requires slight extra equipment cost to accomplish a higher handling speed. If the OPP scenarios for AP application has any changes in coding bandwidth that means, the Eqs. (1) and (2) are repeatedly recomputed for ordering the process.

##### 4.1.3. Prediction and classification Unit

This PCU contains threshold calculator, predictor and classifier. The block diagram of PCU is shown in Fig. 3.

Min.col.R-D cost, Min.adj.R-D cost,  $MB_{count}$ ,  $Frame_{num}$ ,  $TH_{ma_h}$  and  $TH_{ma_l}$  are given to the input for Predictor and Classifier (PC). The internal blocks of the PC are shown in Fig. 4. Based on the threshold value, the input signal compared and produce the three results such as  $M_{CA}$ ,  $M_{CB}$  and  $M_{CC}$ . All these results are given to the CSLA Incrementer for incrementing the input values. The general work process of the PCU is represented as follows. When coding the  $n^{th}$  MB of the  $k^{th}$

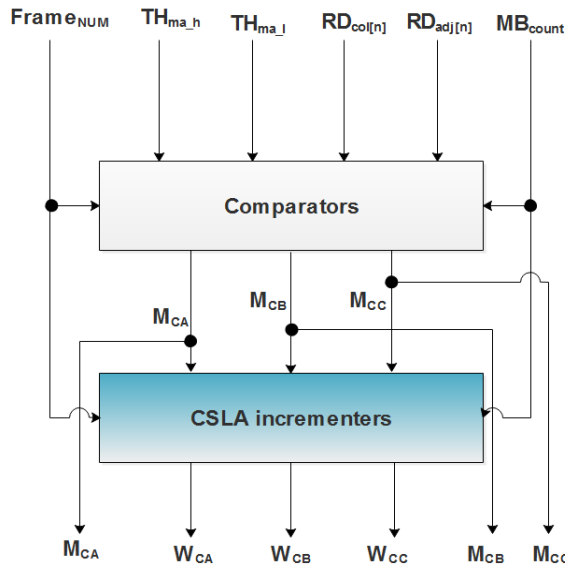


Figure. 4 Block diagram of PC

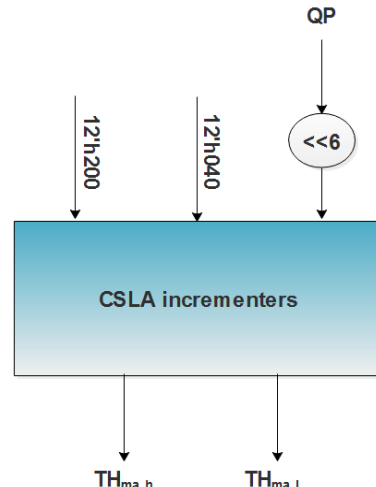


Figure. 5 Block diagram of CSLA Incrementer

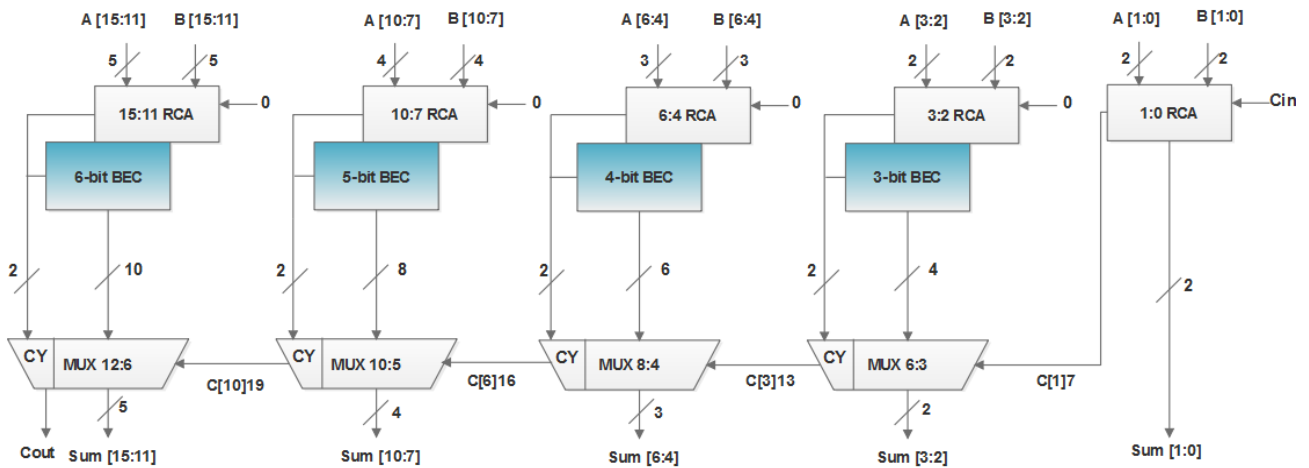


Figure. 6 Block diagram of CSLA design

frame, the movement action of the  $n^{th}$  MB can be predicted with the help of proposed predictor and afterward it grouped with the proposed classifier. During the coding procedure of the  $k^{th}$  frame, the PCU persistently records the three movement action classifications of all the coded MBs in the present coded  $k^{th}$  frame. The block diagram of CSLA Incrementer is shown in Fig. 5.

#### 4.1.4. Carry select adder design

CSLA adder is used to reduce the propagation delay characteristic and for parallel stage operation. The result of the parallel stage operation is taken from several pairs of RCA. The RCAs generate their temporary sum and carry for the CSLA architecture by considering the carry input as zero and one respectively. Fig. 6 shows the block diagram of the CSLA. The CSLA design used in the MFO design for

adding data path instead of the normal adder. It achieves fast arithmetic operation of various data processing approaches. The major objective of this CSLA is to reduce the area in the MFO-CSLA-DBF architecture. The CSLA operates in many computational designs to cut the Carry Propagation Delay (CPD).

The CSLA design uses the Binary-to-Excess Converter (BEC) and Ripple Carry Adder (RCA) with  $C_{in} = 0$  to achieve low area. The CSLA design uses few number of logic gates to derive the BEC logic instead of N-bit Full Adder (FA). The major advantage of BEC logic function is that it used least number of logic gates compared to N-bit FA structure. Fig. 7 shows the operation of the CLSA method, which consist of the four FA designs, those are RCA, BEC and MUX. For example, CSLA operation, in an

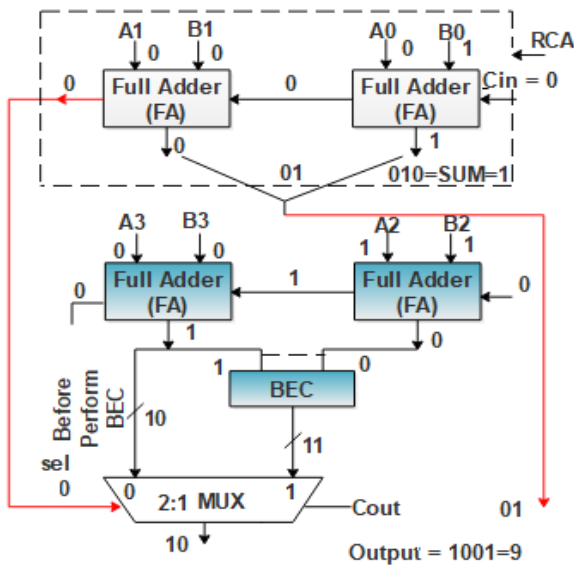


Figure. 7 Operation of the Carry select adder

initial stage,  $A_0 = 0, B_0 = 1, A_1 = 0, B_1 = 0$  are given to the input of the RCA circuit. The group 2 has one 2-b RCA, which has two FA with  $C_{in} = 0$ . The first FA sum is 1 and carry is 0 which is given to the input of second FA. The second full adder sum is 0 and carry also 0, that carry output is given to the input of Mux. While the RCA output is 01.

In the second stage, the  $A_2 = 1, B_2 = 1$  are given to the first FA, and obtained sum is 0 and carry is 1. In second FA,  $A_3 = 0, B_3 = 1$  and first  $FA_{carry} = 1$  values are given to the input of the second FA, which gives sum as 1 and carry as 0. This output of FA is given to the input of the BEC. The main function BEC is one-bit incremental operation. If the input of BEC is 2 (10), then the output will be obtained as 3 (11). The BEC and second stage FA outputs (10) are given to the input of MUX, if selection line is 0 at the time MUX output is 10, if it is 1 at the time MUX output is 11. Finally, the concatenation of MUX output (10) and the first stage output (01) delivers the output as 9 (1001). The input arrival time is less compared to the multiplexer selection input arrival time.

The CSLA Incrementation performed to deliver the final output to the last stages such as MCA, MCB, and MCC, WCA, WCB, and WCC. These outputs are given to the SCBAC for delivering the BCA, BCB, and BCC outputs. Fig. 8 shows the Level Shifter (LS) which is employed to shift a signal voltage range from one voltage domain to another voltage domain, it is needed when the chip is generating the multiple voltage domains. The voltage range is varied for every VLSI circuit design. The

difference in the voltage range can cause unreliable functioning of the destination domain. Therefore, the LS is injected in the voltage domain crossings.

This level shifter is used to allocate the voltage for sub blocks present in the architecture. The voltage supply is represented as 0.9v which shown in Fig. 8. & NMOS and 7 PMOS transistors are used to design the LC design. The length and width of each transistor is represent as 180nm, and 2 micro meter. In this work, 0.9v is considered as supply voltage. In LC-CSLA-ME architecture, entire 0.9v are not required to the respective sub blocks. In that situation, LC is effective one to allocate the supply voltage to respective blocks and it reduces the power consumption of the entire architecture.

## 5. Experimental results and discussion

In this section, the experimental result and discussion of the proposed methodology is detailed effectively and also described about the experimental set-up and performance measure. The performance of the proposed methodology was evaluated by ASIC and FPGA performances.

### 5.1 Experimental setup

The proposed approach experimented using 4GB RAM with 3.30 GHz, i3 processor and 500GB hard disk. The architecture has been implemented using Verilog language. MATLAB tool used for reading the input frame and shows the ME output. Modelsim 10.5 tool is used to write a Verilog code and verifying the timing diagram. Xilinx 14.4 is used for evaluating FPGA performances like LUT, flip flop, slices, and frequency. The Cadence RTL compiler used to calculate ASIC performances like area, power, and delay. 180nm and 45nm technologies are used to analyse the ASIC performances. Cadence AMS design is used for reading the Verilog code in the analog design.

### 5.2 Discussion

The ASIC performance of the different methods is tabulated in Table 1. This table presented a comparison of the Existing-I [17], existing-II [19], existing-III [20], existing-IV [21], and LC-CSLA-ME. Normal adder has used to perform the ME operation [21]. Optimal CSLA adder is used in LC-CSLA-ME method.



The comparison of ASIC performances is tabulated in Table 1. Many architectures have been used previously. Here, all the 4 existing methods are implemented and the results are tabulated. All the methods are implemented in the cadence RTL compiler with 180nm and 45nm technology. From this table, it's clear that LC-CSLA-ME provided better performances in area, power, and delay when compare to existing architectures.

The comparison graph of area, power, and delay is presented in Fig. 9, Fig.10, and Fig.11. All these graphs show the comparison of performance of both 180nm and 45nm technology. All the ASIC performances are reduced in LC-CSLA-ME due to the LC and CSLA.

The performances of FPGA tabulated in Table 2. In this table, Virtex 4 and Virtex 5 devices are used to evaluate LUT, flip flop, slices, frequency, and Bonded IOB. These values show that LC-CSLA-ME architecture occupy less FPGA performance parameters.

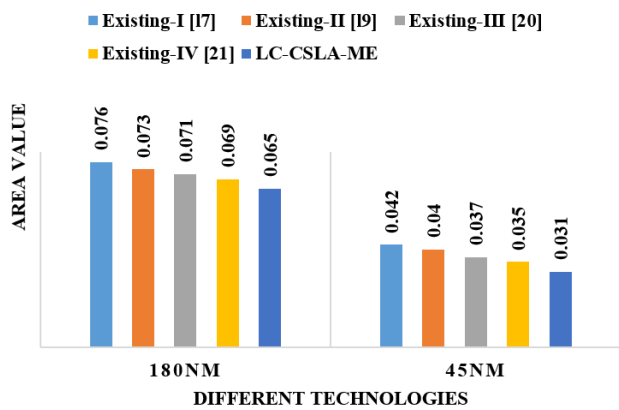


Figure. 9 Comparison of area performance of different methods

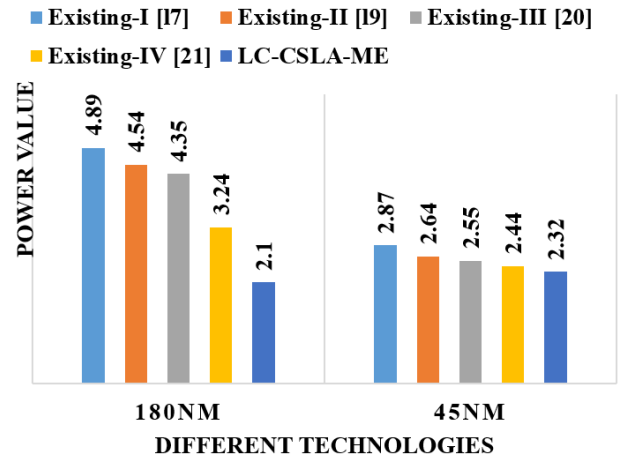


Figure. 10 Comparison of power performance of different methods

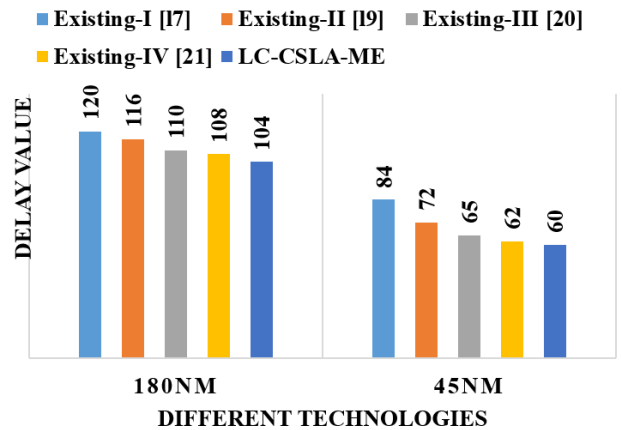


Figure. 11 Comparison of delay performance of different methods

Table 2. Comparison of FPGA performances for different methods

Devices	Method	LUT	Flip flop	Slices	Frequency (MHz)	Bonder IOB
Virtex 4	Existing-I [17]	3147	457	1854	344.4	284
	Existing-II [19]	3077	374	1702	100	252
	Existing-III [20]	3049	365	1688	400	232
	Existing-IV [21]	3021	350	1577	1100	221
	LC-CSLA-ME	2978	336	1325	1148	215
Virtex 5	Existing-I [17]	1578	289	987	241.4	178
	Existing-II [19]	1564	264	945	89.8	164
	Existing-III [20]	1498	252	899	347.2	161
	Existing-IV [21]	1455	248	845	1047.2	157
	LC-CSLA-ME	1347	234	789	1098.5	151



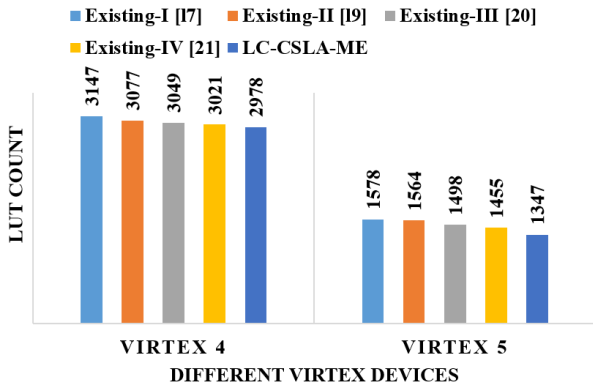


Fig.12. Comparison of LUT for different methods

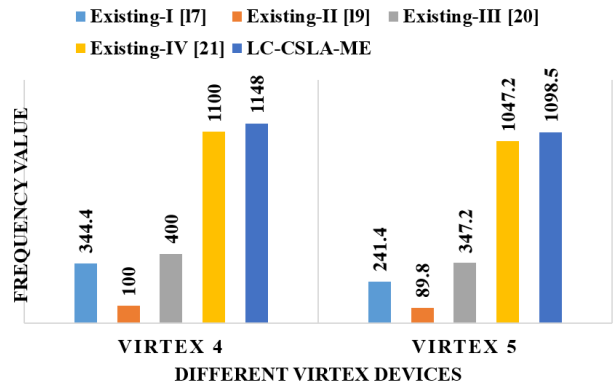


Figure. 15 Comparison of frequency for different methods

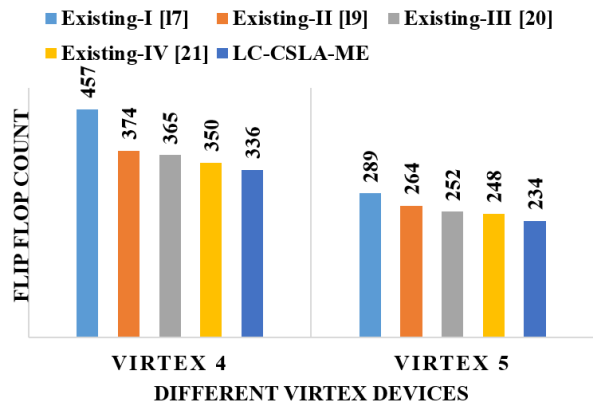


Figure. 13 Comparison of flip flop for different methods

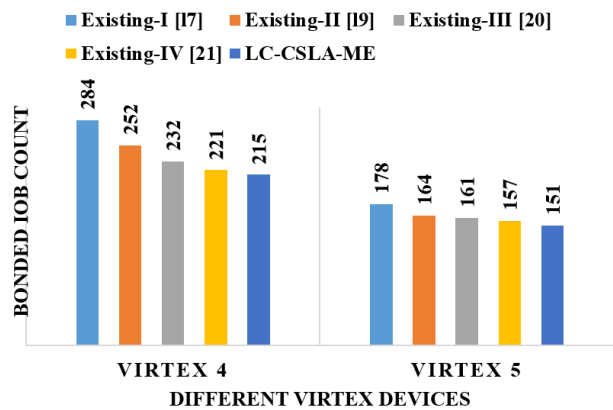


Figure. 16 Comparison of bonded IOB for different methods

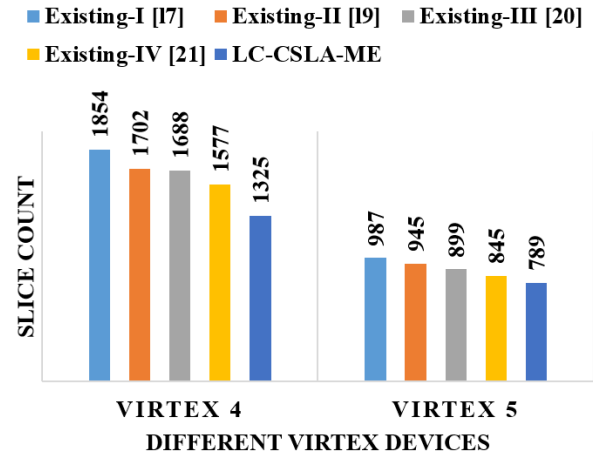


Fig. 14 Comparison of slices for different methods

The comparison graph of LUT, Flip flop, slices, frequency and bonded IOB are shown in Figs. 12, 13, 14, 15, and 16. All these graphs show the comparison of performance of both Virtex 4 and Virtex 5 devices. The hardware utilizations evaluated from this performance of FPGA. The delay has been evaluated in ASIC performances and frequency in FPGA performances. It clearly tells that the speed of the entire architecture has been improved. The comparison of CTS for different data sets and specifications are given in Tables 3 and 4. Different data sets such as four people, Sunflower, Johnny, Kristen and Sara, Tractor, Vidyo1, Blue sky, Vidyo3, Sation2, Vidyo4, and Rush Hour are used to analyze the Operating Performance Point (OPP).

Table 3. Comparison of CTS for different data sets


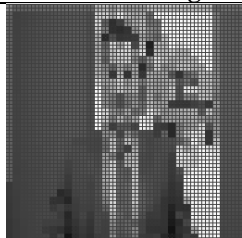



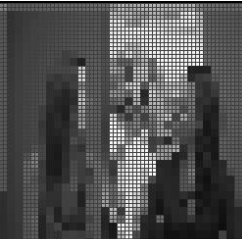






Sequence	Method	OPP1	OPP2	OPP3	OPP4	OPP5
Four people	Existing-IV [21]	32.139	52.053	64.694	72.978	78.609
	LC-CSLA-ME	33.471	53.214	65.124	73.654	79.465
Sunflower	Existing-IV [21]	31.350	52.252	64.604	72.955	78.827
	LC-CSLA-ME	32.145	53.999	65.214	73.245	79.254
Johnny	Existing-IV [21]	31.405	51.190	63.951	72.110	78.014
	LC-CSLA-ME	31.987	52.147	64.897	73.654	79.621
	Existing-IV [21]	31.848	51.990	64.508	72.611	78.330

Kristen and sara	LC-CSLA-ME	31.925	52.165	65.321	73.254	79.654
Tractor	Existing-IV [21]	35.926	54.710	66.652	74.582	80.077
	LC-CSLA-ME	36.214	55.694	67.584	75.485	81.258
Vidyo1	Existing-IV [21]	32.132	51.731	64.500	72.855	78.643
	LC-CSLA-ME	33.214	52.364	65.321	73.654	79.621
Blue sky	Existing-IV [21]	36.106	56.145	67.381	75.293	80.650
	LC-CSLA-ME	36.845	57.321	68.654	76.987	81.364
Vidyo3	Existing-IV [21]	33.810	53.714	65.946	73.957	80.042
	LC-CSLA-ME	34.578	54.698	66.120	74.654	81.021
Sation2	Existing-IV [21]	31.906	52.605	64.657	72.873	78.633
	LC-CSLA-ME	32.147	53.333	65.120	73.524	79.027
Vidyo4	Existing-IV [21]	33.227	52.541	66.172	73.215	78.925
	LC-CSLA-ME	34.369	53.987	67.482	74.258	79.665
Rush Hour	Existing-IV [21]	32.845	51.369	64.421	72.317	78.080
	LC-CSLA-ME	33.987	52.987	65.547	73.694	79.645
Average	Existing-IV [21]	32.972	52.755	65.226	73.250	78.985
	LC-CSLA-ME	34.021	54.316	67.021	75.027	80.071

Table 4. Comparison of specification for different methods

Parameters	Existing-IV [21]	LC-CSLA-ME
Voltage	0.9	0.9 (LC)
Technology	90	180,45
Max.Res	1280x720	1280x720
DVFS aware capability	Yes	yes
Bandwidth control	5,4,3,2,1	5,4,3,2,1
Logic gate count	0.816	0.798
Average bandwidth saving	56.08	54.02

Table 5. Comparison of Motion estimation for different data sets

Sequence	Input frame	Sub-Blocking	Motion estimation	Frame retrieval
Johnny				
Kristen and Sara				
Vidyo1				

The comparison of the ME for different datasets is given in Table 5 which shows the Johnny, Kristen & Sara and Vidyo 1 datasets operation. These tables include the input frame, sub blocking frame, ME frame, and retrieval frame. This table clearly shows that the ME worked effectively with less hardware utilization.

## 6. Conclusion

The ME architecture has been designed effectively to reduce the hardware utilization of LC-CSLA-ME. In this work, LC-CSLA-ME architecture is designed to detect the motion estimation for different datasets such as four people, Sunflower, Johnny, Kristen, sara, Tractor, Vidyo1, Blue sky, Vidyo3, Sation2, Vidyo4, and Rush Hour. The Proposed LC-CSLA-ME architecture is developed using Verilog code. The main contribution of this work is LC and CSLA adder which is used for reducing the power and area. In 180nm technology, LC-CSLA-ME architecture occupied 0.065mm<sup>2</sup> area, 2.10mW power, and 104ms delay. In Virtex 4, LC-CSLA-ME architecture occupied 2978 LUT, 336 flip flop, 1325 slices, 1148MHz frequency, and 215 bonded IOB. The CTS performance of LC-CSLA-ME architecture improved compared to existing architecture. In the future, various motion estimation algorithms will be designed by using optimal adder like Carry Look Ahead Adder, and Carry Incrementer Adder.

## References

- [1] H. Liang and T. Morie, "A motion detection model inspired by hippocampal function and its applications to obstacle detection", *Neurocomputing*, Vol.129, No.7, pp.59-66, 2014.
- [2] L. Lu, J.V. McCanny, and S. Sezer, "Reconfigurable system-on-a-chip motion estimation architecture for multi-standard video coding", *IET Computers & Digital Techniques*, Vol.4, No.5, pp.349-364, 2010.
- [3] M. Kthiri, H. Loukil, A. B. Atitallah, P. Kadionik, D. Dallet, and N. Masmoudi, "FPGA architecture of the LDPS Motion Estimation for H. 264/AVC Video Coding", *Journal of Signal Processing Systems*, Vol.68, No.2, pp.273-285, 2012.
- [4] J. Vasiljevic and A. Ye, "Analysis and architecture design of scalable fractional motion estimation for H. 264 encoding", *Integration, the VLSI Journal*, Vol.45, No.4, pp.427-438, 2012.
- [5] G. He, D. Zhou, Y. Li, Z. Chen, T. Zhang, and S. Goto, "High-throughput power-efficient VLSI architecture of fractional motion estimation for ultra-HD HEVC video encoding", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.23, No.12, pp.3138-3142, 2015.
- [6] W. Elhamzi, J. Dubois, J. Miteran, M. Atri, B. Heyrman, and D. Ginjac, "Efficient smart-camera accelerator: A configurable motion estimator dedicated to video codec", *Journal of Systems Architecture*, Vol.59, No.10, pp.870-877, 2013.
- [7] W. Elhamzi, J. Dubois, J. Miteran, M. Atri, B. Heyrman, and D. Ginjac, "An efficient low-cost FPGA implementation of a configurable motion estimation for H. 264 video coding", *Journal of Real-Time Image Processing*, Vol.9, No.1, pp.19-30, 2014.
- [8] H. Ho, R. Klepko, N. Ninh, and D. Wang, "A high performance hardware architecture for multi-frame hierarchical motion estimation", *IEEE Transactions on Consumer Electronics*, Vol.57, No.2, pp.794 – 801, 2011.
- [9] G. Botella, U. Meyer-Baese, A. GarcíA, and M. Rodríguez, "Quantization analysis and enhancement of a VLSI gradient-based motion estimation architecture", *Digital Signal Processing*, Vol.22, No.6, pp.1174-1187, 2012.
- [10] M. Akin, Z. Cetin, B. Ozcan, B. Erbagci, and M. Hamzaoglu, "An adaptive bilateral motion estimation algorithm and its hardware architecture", *IEEE Transactions on Consumer Electronics*, Vol.58, No.2, pp.712-720, 2012.
- [11] M. Kraft and A. Kasiński, "Implementation of the Robot Ego-Motion Estimation Algorithm with FPGA Circuits", *IFAC Proceedings Volumes*, Vol.43, No.24, pp.16-21, 2010.
- [12] P. Nalluri, L.N. Alves, and A. Navarro, "High speed SAD architectures for variable block size motion estimation in HEVC video coding", In: *Proc. of International. Conf. on Image Processing*, pp.1233-1237, 2014.
- [13] F. Barranco, M. Tomasi, J. Diaz, M. Vanegas, and E. Ros, "Parallel architecture for hierarchical optical flow estimation based on FPGA", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.20, No.6, pp.1058-1067, 2012.
- [14] S.M. Basha and M. Kannan, "Design and implementation of low-power motion estimation based on modified full-search block motion estimation", *Journal of Computational Science*, Vol.21, pp.327-332, 2017.

- [15] A. Rodriguez and F. Moeno, "Evolutionary computing and particle filtering: A hardware-based motion estimation system", *IEEE Transactions on Computers*, Vol.64, No.11, pp.3140-3152, 2015.
- [16] R. Mukherjee, B. Biswas, I. Chakrabarti, P.K. Dutta, and A. K. Ray, "Efficient VLSI design of adaptive rood pattern search algorithm for motion estimation of high definition videos", *Microprocessors and Microsystems*, Vol.45, pp.105-114, 2016.
- [17] P. P. Shiju, I. Chakrabarti, R. Viridi, and S. Wasnik, "VLSI architecture for fixed mesh based deformable motion estimation using ARPS algorithm", *Microprocessors and Microsystems*, Vol.59, pp.92-102, 2018.
- [18] B. Biswas, R. Mukherjee, I. Chakrabarti, P.K. Dutta, and A. K. Ray, "Efficient architecture of adaptive rood pattern search technique for fast motion estimation", *Microprocessors and Microsystems*, Vol.39, No.3, pp.200-209, 2016.
- [19] S.P. Puthenpurayil, I. Chakrabarti, R. Viridi, and H. Kaushik, "Very large scale integration architecture for block-matching motion estimation using adaptive rood pattern search algorithm", *IET Circuits, Devices & Systems*, Vol.10, No.4, pp.309-316, 2016.
- [20] G. Pastuszak and M. Trochimiuk, "Algorithm and architecture design of the motion estimation for the H. 265/HEVC 4K-UHD encoder", *Journal of Real-Time Image Processing*, Vol.12, No.2, pp.517-529, 2016.
- [21] J.H. Hsieh and H.R. Wang, "VLSI Design of an ML-Based Power-Efficient Motion Estimation Controller for Intelligent Mobile Systems", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.26, No.2, pp.262-271, 2018.