IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197 Volume 1, Issue 6 (Mar. – Apr. 2013), PP 10-18
www.iosrjournals.org

Power Efficient Sum of Absolute Difference Algorithms for video Compression

D.V. Manjunatha[1], G. Sainarayanan[2]

[1]*(Department of ECE, Don Bosco Institute of Technology/ Visveswaraya Technological University, India)*
[2]*(Department of EEE, New Horizon College of Engineering/ Visveswaraya Technological University, India)*

Abstract : *Video Compression (VC) is one of the resource hungry key element in video communication and is commonly achieved using Motion Estimation (ME). In this paper we proposed power efficient one bit full adder and one of the simple and easy metric called Sum of Absolute Difference (SAD) for estimating the motion vectors in motion estimation. SAD is primarily used to detect motion in the video sub system. Here we proposed power efficient 4X4 and 8X8 SAD architectures. The proposed 4X4 SAD proves that 29%, 63.23% and 61.31% improvement in leakage power, dynamic power and total power respectively as compared with existing 4X4 SAD. Similarly the proposed 8X8 SAD which proves that 57%, 46.16% and 46.78% improvement in Leakage Power (LP), Dynamic Power (DP) and Total Power (TP) respectively as compared with that of existing 8X8 SAD at the gate level. The designs are implemented in ASIC methodology using cadence tools.*

Keywords - *SAD, ME, VC, CSA, DSP, LP, DP, TP, FPGA etc.*

I. Introduction

Many recent multimedia applications of digital video systems such as video conferencing, video-on-demand, video-phone, distance learning and digital TV, object tracking and many more have become popular products because of their convenience. Such applications require video compression with ever higher compression ratios, better visual quality and high bandwidth. The high efficiency video compression commonly uses the efficient hardware architectures. In general the development of hardware architectures are designed to form the integrated circuits which allows parallel processing of data from various sub blocks of the architectures, however hardware architectures suffers from limitations such as algorithm flexibility due to timing dependencies, which arises from the dataflow of various blocks of the architecture. Thus the development of good architecture of video codec in integrated circuits is very important. The customization of video codec is the video compression in the modern state of the art real time Digital Signal Processing (DSP) systems.

Video compression is one of the techniques in video processing system to reduce resource usage. The two primary challenges addressed during video compression are:

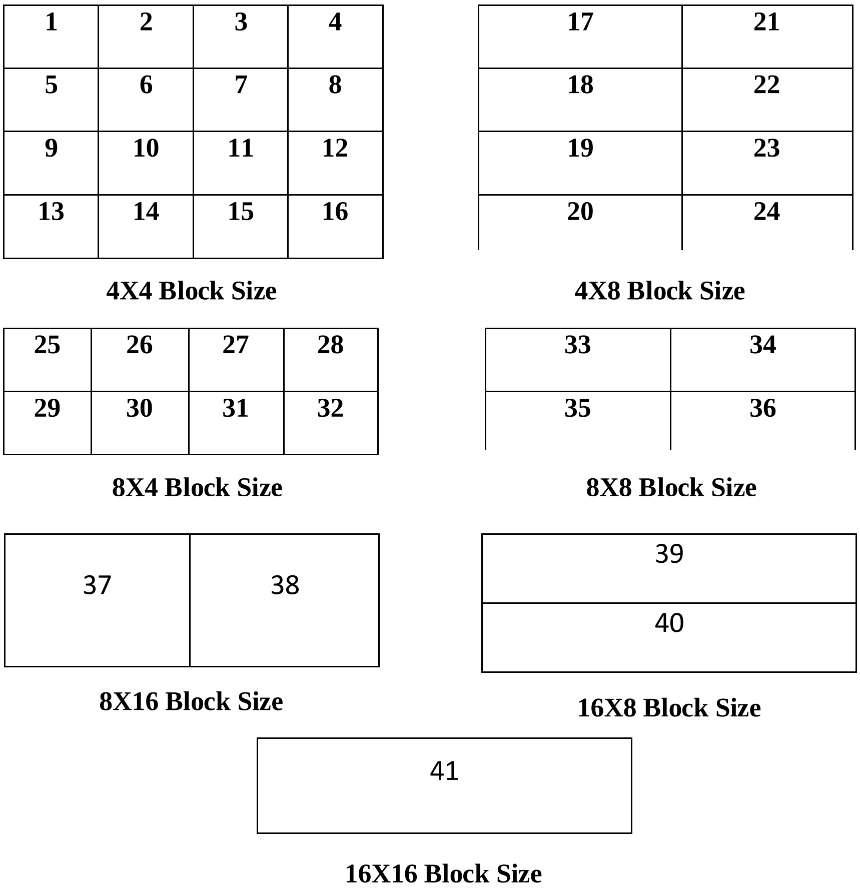
- Limited Network bandwidth.
- Limited Storage capacity.

Hence the two important metrics of a video encoder are low computational complexity & low power hardware implementation. Present day world, compression ratio plays the major role in the field of video processing. The motion in the video scene will reduce the efficiency of the compression ratio. Hence the motion estimation field has seen the highest research topic and interested issue in the past a few decades. In short the motion estimation means the estimation of the displacement (or velocity) of image structures from one frame to another in a time sequence of 2-D images of the video in order to achieve video compression in video coding.

The efficiency of the compression ratio can be increased by exploiting the similarities between the video frames. The simple metric system is the SAD algorithm, where the absolute differences between the corresponding elements are added up.

There are varieties of video coding standards in the video processing systems; the modern video coding standard is H.264/AVC. This coding standard uses the Variable Block Size Motion Estimation (VBSME), in this standard; the computation requirements are much higher than the previous coding standards such as H.263/ MPEG-IV. In H.264/AVC, each picture frame is divided into many macro blocks.

Each macro block is further divided into seven different sub-block sizes they are 4×4, 4×8, 8×4, 8×8, 8×16, 16×8 and 16×16 as shown in figure below



4X4 Block Size

4X8 Block Size

8X4 Block Size

8X8 Block Size

8X16 Block Size

16X8 Block Size

16X16 Block Size

Figure 1: Different block sizes of motion estimation in h.264.

In this work we first identify the low power architecture at the level of 1bit addition (full adder) here five full adder architectures are synthesized based on, which architecture is giving the low power solution, such a full adder is used in the ripple carry parallel adder and the carry save adders, then these adders are used in basic sub block sizes of VBSME such as 4X4 and 8X8 are used in SAD architectures. Using these two SAD architectures remaining SAD architectures can be obtained for achieving variable block size motion estimation in video coding (compression). There are many tradeoffs encountered during the design of such modules (will be discussed in results section).

II. RELATED WORK

Several methods of finding the motion vectors have been presented in the literature, where there is a tradeoff between the power dissipation, area and the latency in the optimality of hardware implementation. The work presented by [1-6] shows that motion estimation aims at reducing the temporal redundancy between successive frames in a video sequence. Innovation has put much emphasis on improving the video-coding giving rise to new standard H.264/AVC [7, 8].

The coding efficiency in this new standard is increased to about 40–60% as compared with the Motion Picture Experts Group (MPEG)-2 and H.263 standards. Chen et al. [9] have presented H.264/AVC encoder which employs 1024 SAD processing Elements (PE) which uses 305k gates. Vanne et al. [10] have proposed SAD architecture and compared much architecture in terms of area and delay.

Yufei et al. [11] proposed the SAD architecture with 1st and 2nd stage pipeline; Stephan Wong, et al. [13] describes the parallel hardware implementation of the SAD operation in field-programmable gate arrays (FPGAs). A novel SAD16 unit which performs a 16 x1 SAD operation is proposed by C Hisham, et al [14], the work done in [9-13] presented the SAD architectures in terms of gate count and delay optimization.

The work by S. Kappagantula, et al. [15] presents that the two types of redundancies can be reduced using the predictive coding but more compensation can be reached by using it together with motion estimation. The work by S. Vassiliadis, et al. [16] proposes that the sum of absolute differences algorithm is used in determining the motion vectors in video coding. This paper is organized as follows: Section 3 describes the various existing gate level realizations and the proposed low power 1-bit binary full adder. Section 4 describes the higher order adder architectures in the adder exploration. Section 4 gives the details about the existing and proposed SAD architectures at 4X4 (16 samples) and 8X8 (64 input samples) level. Section 5 describes the results and discussion, finally we conclude in section 6.

III. 1BIT ADDER ARCHITECTECTURES

The 1bit adder is the most basic elements of many critical data paths of the digital arithmetic circuits and digital signal processors. A 1 bit full adder is basically a combinational logic circuit, which performs binary addition operation on 3 single bit binary numbers and produces two outputs called sum and the carry. There are many efficient full adder architectures in the literature namely i) 2 EXNOR gate and 1 MUX architecture ii) 1 EXNOR and 2 MUX based Full adder iii) EXOR, AND & OR gate Architecture iv) EXOR and Nand Gates Architecture and v) The proposed low power NAND, AND & OR Gate architecture.

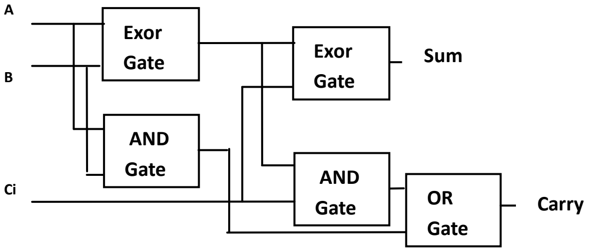


Figure 2: EXNOR & MUX architecture of Full Adder

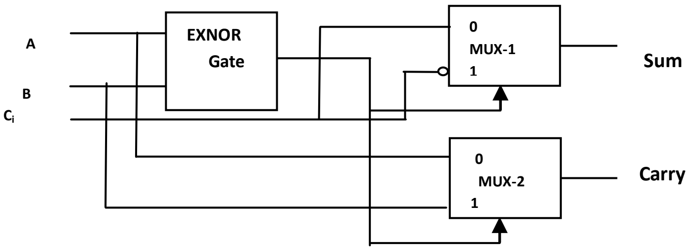


Figure 3: 1 XNM EXNOR & MUX based Full adder

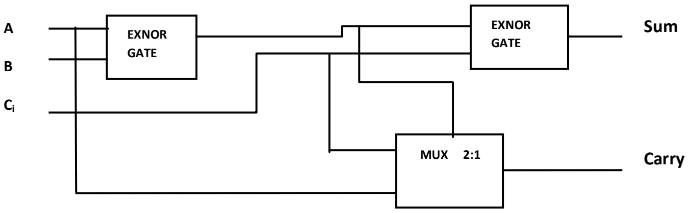


Figure 4: EXOR, AND & OR gate Full adder

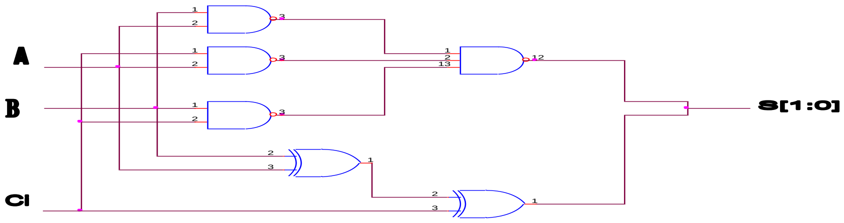


Figure 5: EXOR and Nand gates Full Adder

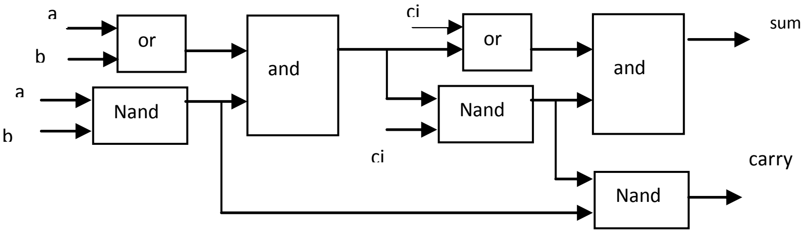


Figure 6: Proposed NAND, AND & OR gate full adder architecture

Out of the above five full adder architectures the proposed NAND, AND & OR gate architecture of full adder consumes low power, in this proposed architecture the carry generation is faster and the overall power consumption is made to below.

IV. ADDER EXPLORATION

The multi bit binary addition is the most important operation used in arithmetic operation on digital video / image processors. Hence multi bit adders are the most important blocks in building digital systems. The key challenge addressed in this work is the low power multi bit adders.

Various adder architectures have been proposed in the literature covering wide range of performance goals, but the suitability of the adder architectures for cell based design and hardware synthesis has been the very important for the low power addition, based upon the latency there are three adder architectures they are

4.1 Ripple Carry: It is the simplest architecture for an n-bit adder, intermediate carries are generated sequentially. It has smallest area, longest delay & consumes lowest power when two multi bit binary numbers are added. The ripple carry adder architecture is as shown in figure 7.

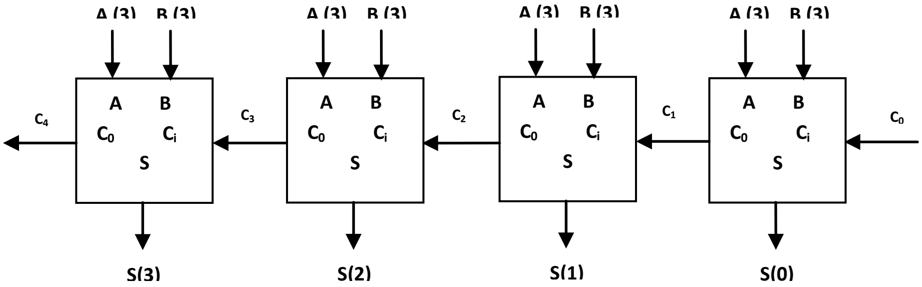


Figure 7: 4 bit ripple carry adder.

4.2 Carry Look Ahead: Here the carry is generated in parallel to reduce the processing delay. Hence it is faster than ripple carry architecture, and it consumes more area and power

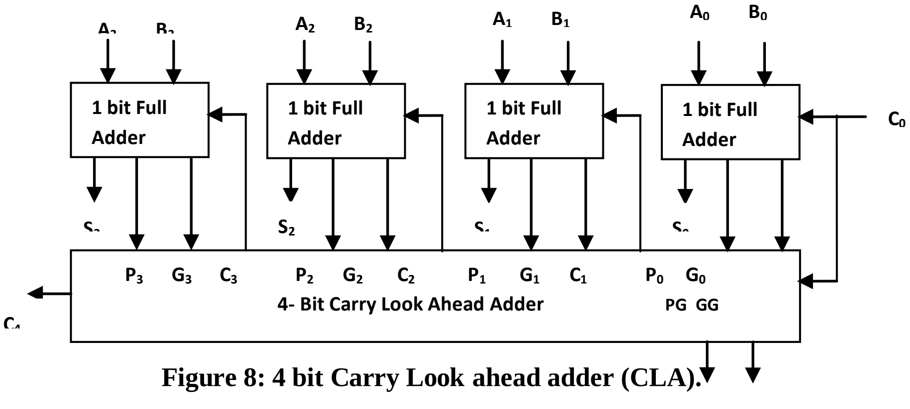


Figure 8: 4 bit Carry Look ahead adder (CLA).

4.3 Carry Save Addition: when multi bit addition is required then the low power adder called the Carry save adder is used which give very low power. The carry save adder circuit is as shown below

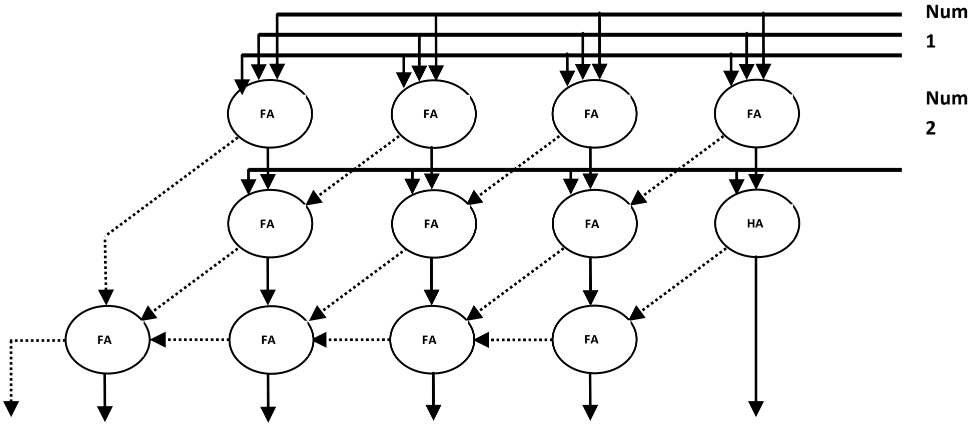


Figure 9: Carry Save Adder (CSA).

The multi bit binary addition is the most important operation used in arithmetic operation on digital video / image processors. Hence multi bit adders are the most important blocks in building digital systems. The key challenge addressed in this work is the low power multi bit adders.

V Sad Architectures

The sum of absolute differences (SAD) is the most repeated operation in block matching motion estimation subsystem. SAD algorithm is used for measuring the similarities between the images by calculating the absolute differences between the pixels of the image (Template image) and their corresponding ones (search image) in the macro block and then these differences are added up to result in the similarity block. It requires only two basic mathematical operations addition & shifting. The SAD algorithm is the simplest metric which considers all the pixels in the block for computation and also separately, which makes its implementation easier and parallel, due to its simplicity this algorithm is one of the fastest and can be used widely in block motion estimation and object recognition. This paper proposes the 4X4, 8x8 sum of absolute differences algorithm used in motion estimation of video compression. The architecture is able to perform a full motion search on integral multiples of 4X4 and 8x8 blocks sizes. The block and hierarchy diagrams of Sum of Absolute Difference are shown in figures 10 and 11 respectively.

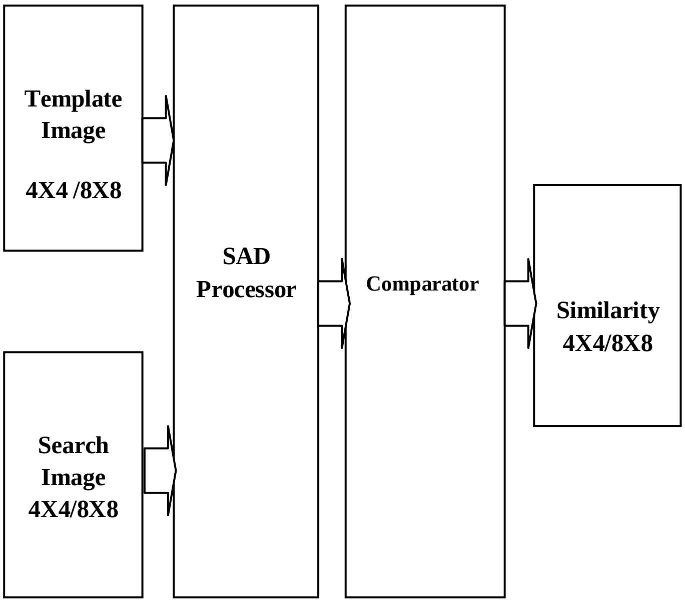


Figure 10: Block Diagram of 8X8 Sum of Absolute Difference block.

The sum of absolute difference architecture hierarchy is as shown below:

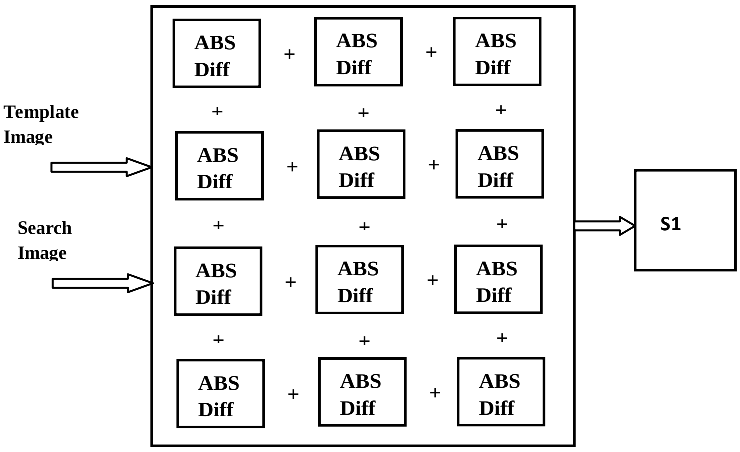


Figure 11: Hierarchy of sum of absolute difference block

Typical steps involved in fully parallel sum of absolute difference architecture are:

- Perform absolute difference of all the pixels (of a block of video).
- Perform sum of all the absolute differences.
- Select block with minimum difference value.

VI Results and Discussion

In this paper both 4X4 and 8X8 sum of absolute difference algorithms are implemented in three different ways they are

1. Implementation of SAD using normal existing ripple carry adder (which uses existing NAND and EXOR gates full adder).
2. Implementation of SAD using proposed ripple carry adder (which uses proposed NAND, AND & OR gate architecture for full adder).
3. Implementation of SAD using proposed carry save adder (which uses proposed NAND, AND & OR gate architecture for full adder).

The above SAD architectures were implemented in ASIC methodology. The architectures are modeled using verilog coding, functionally verified using modelsim and synthesized using RTL compiler. The results are tested using 180nm technology library of cadence EDA tools, the results are presented at 1 bit adder, 4X4 SAD and 8X8SAD levels as shown below:

Particulars	LP nW	DP nW	Total Power nW	Delay T in Ps	Area in Sq Microns
Full Adder using EXOR & Nand gates existing	96.144	492.446	588.590	157	24
proposed Full Adder using EXOR, AND & OR Gates	51.390	314.024	365.414	276	19

Table1: Full adder synthesis results Using 180 nano meter Technology

Particulars	LP µW	DP µW	Total Power µW	Delay T in Ps	Area in Sq microns
4X4 SAD using normal existing ripple carry adder	26.13	440.94	467.07	01561	17283
4X4 SAD using proposed ripple carry adder	18.55	162.13	180.68	02666	5492
4X4 SAD using proposed carry save adder	25.15	500.15	525.30	04490	6397

Table2: 4X4 SAD synthesis results Using 180 nano meter Technology.

Particulars	**LP µW**	**DP µW**	**Total Power µW**	**Delay T in Ps**	**Area in Sq microns**
8X8 SAD using normal existing ripple carry adder	119.61	1980.34	2099.95	5059	30643
8X8SAD using proposed ripple carry adder	85.23	1738.71	1823.94	5473	26080
8X8 SAD using proposed carry save adder	51.31	1066.23	1117.54	4954	12999

Table3: 8X8 SAD synthesis results Using 180 nano meter Technology.

Note: LP = Leakage Power; DP = Dynamic Power; T = Delay; A = Area;

The synthesis snap shot diagrams of both proposed 4X4 SAD and 8X8 SAD are as shown below

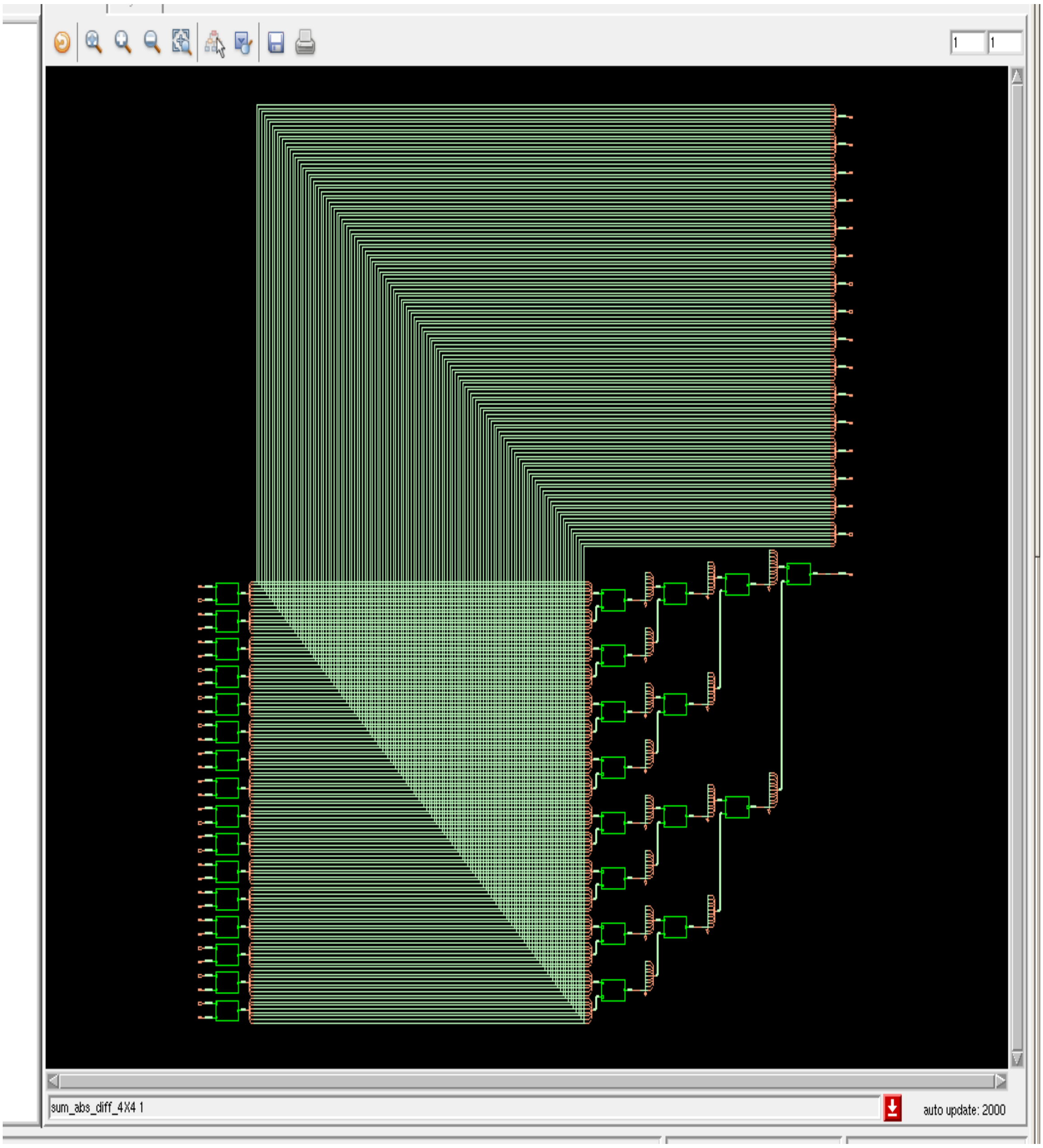
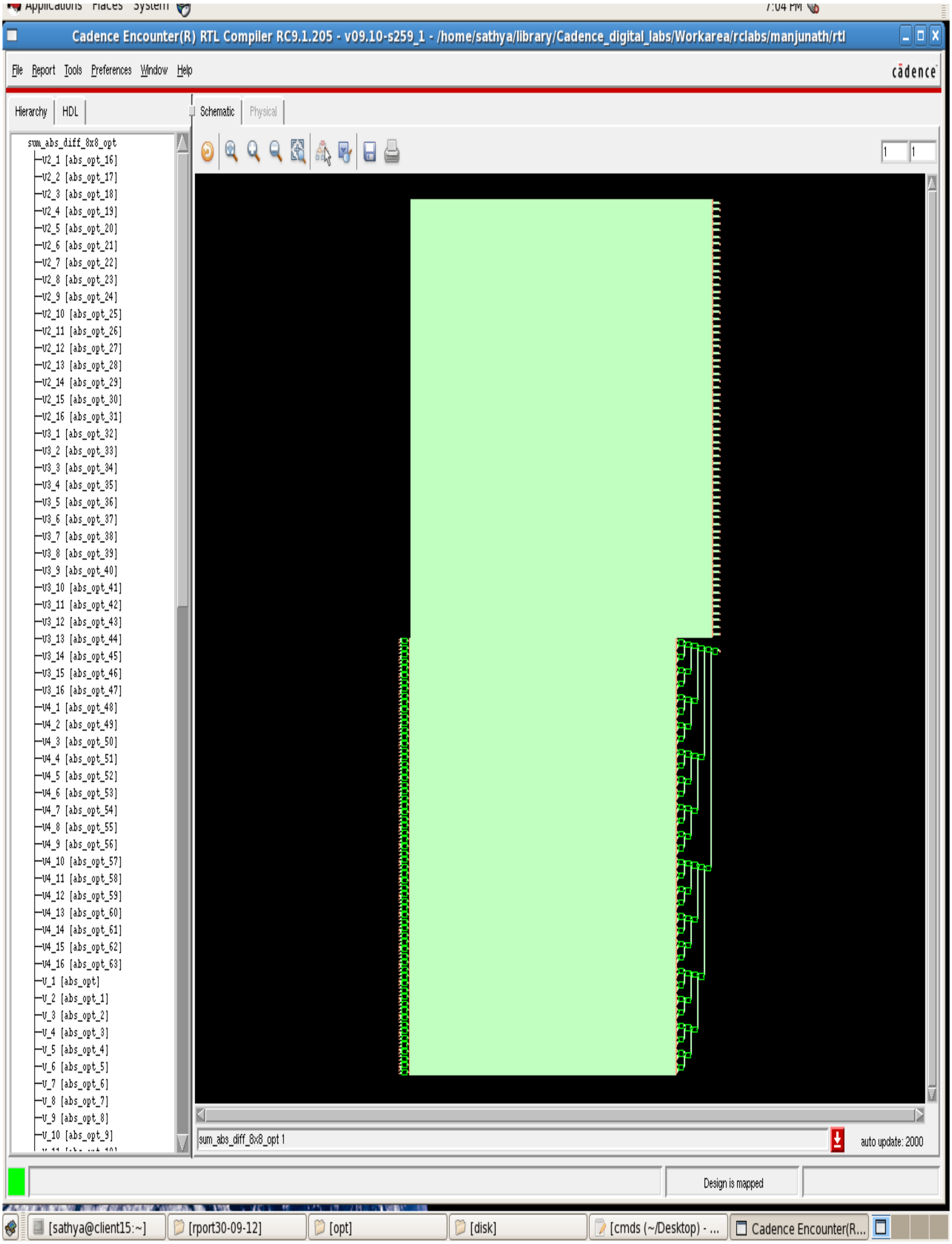


Figure12: Synthesis snapshot of proposed 4X4 Sum of Absolute Difference


Applications Places System
7:04 PM
Cadence Encounter(R) RTL Compiler RC9.1.205 - v09.10-s259_1 - /home/sathya/library/Cadence_digital_labs/Workarea/rclabs/manjunath/rtl
File Report Tools Preferences Window Help
cadence
Hierarchy HDL
Schematic Physical
1 1
sum_abs_diff_8x8_opt 1
auto update: 2000
Design is mapped
[sathya@client15:~]
[rport30-09-12]
[opt]
[disk]
[cmds (~/Desktop) - ...
Cadence Encounter(R...

```
sum_abs_diff_8x8_opt
├U2_1 [abs_opt_16]
├U2_2 [abs_opt_17]
├U2_3 [abs_opt_18]
├U2_4 [abs_opt_19]
├U2_5 [abs_opt_20]
├U2_6 [abs_opt_21]
├U2_7 [abs_opt_22]
├U2_8 [abs_opt_23]
├U2_9 [abs_opt_24]
├U2_10 [abs_opt_25]
├U2_11 [abs_opt_26]
├U2_12 [abs_opt_27]
├U2_13 [abs_opt_28]
├U2_14 [abs_opt_29]
├U2_15 [abs_opt_30]
├U2_16 [abs_opt_31]
├U3_1 [abs_opt_32]
├U3_2 [abs_opt_33]
├U3_3 [abs_opt_34]
├U3_4 [abs_opt_35]
├U3_5 [abs_opt_36]
├U3_6 [abs_opt_37]
├U3_7 [abs_opt_38]
├U3_8 [abs_opt_39]
├U3_9 [abs_opt_40]
├U3_10 [abs_opt_41]
├U3_11 [abs_opt_42]
├U3_12 [abs_opt_43]
├U3_13 [abs_opt_44]
├U3_14 [abs_opt_45]
├U3_15 [abs_opt_46]
├U3_16 [abs_opt_47]
├U4_1 [abs_opt_48]
├U4_2 [abs_opt_49]
├U4_3 [abs_opt_50]
├U4_4 [abs_opt_51]
├U4_5 [abs_opt_52]
├U4_6 [abs_opt_53]
├U4_7 [abs_opt_54]
├U4_8 [abs_opt_55]
├U4_9 [abs_opt_56]
├U4_10 [abs_opt_57]
├U4_11 [abs_opt_58]
├U4_12 [abs_opt_59]
├U4_13 [abs_opt_60]
├U4_14 [abs_opt_61]
├U4_15 [abs_opt_62]
├U4_16 [abs_opt_63]
├U_1 [abs_opt]
├U_2 [abs_opt_1]
├U_3 [abs_opt_2]
├U_4 [abs_opt_3]
├U_5 [abs_opt_4]
├U_6 [abs_opt_5]
├U_7 [abs_opt_6]
├U_8 [abs_opt_7]
├U_9 [abs_opt_8]
├U_10 [abs_opt_9]
```

Figure 13: Synthesis snapshot of proposed 8X8 Sum of Absolute Difference

Looking at the synthesis results tabulated in the above tables, the following are the salient features of the paper.

1. The proposed 4X4 SAD using ripple carry adder structure proves that 29% improvement in leakage power dissipation, 63.23% improvement in dynamic power dissipation and 61.31% improvement in the total power dissipation as compared with existing 4X4 SAD using ripple carry adder at the gate level
2. In 8X8 SAD using proposed ripple carry adder about 28.70% improvement in leakage power dissipation and 12.20% improvement in dynamic power dissipation and 13.14% improvement in the total power dissipation as compared with 8X8 SAD using existing ripple carry adder.

3. In 8X8 SAD using proposed carry save adder about 39.79% improvement in leakage power dissipation and 38.73% improvement in dynamic power dissipation and 38.67% improvement in the total power dissipation as compared with 8X8 SAD using proposed ripple carry adder.
4. 8X8 SAD using proposed carry save adder proves that 57% improvement in leakage power dissipation and 46.16% improvement in dynamic power dissipation and 46.78% improvement in the total power dissipation as compared with 8X8 SAD using existing ripple carry adder.

VII Conclusion

In this paper we implemented the existing and the proposed 4X4 and 8X8 sum of absolute differences. Here the low power 1 bit Full adder Cell is proposed and is used in the design of sum of absolute difference algorithms. The SAD designs are implemented using ripple carry adder and carry save adder structures, the designs are implemented using 180 nm technology in ASIC flow, the simulations are done using modelsim and the synthesis is done using cadence-RC compiler, the implemented designs concludes that

(i) The 4X4 SAD using proposed ripple carry adder is the area and power efficient architecture and
(ii) The 8X8 SAD using proposed carry save adder is the area and the power efficient architecture as compared with both 8X8 SAD using existing ripple carry adder and the 8X8 SAD using proposed ripple carry adder structures.

At the gate level without optimization and if we try to optimize the same using power optimization techniques further improvement can be achieved.

Acknowledgement

The authors would like to thank Prof Venkatesvarlu and Prof. Siva Yellampalli of UTL technologies for their support in the lab, and especially the first author is thankful to the management of Don Bosco Institute of Technology, Bangalore for their constant encouragement and support.

REFERENCES

[1] Yankang Wang, Kuroda H, "Hilbert scanning search algorithm for motion estimation," *IEEE transactions on circuits and systems for video technology, vol. 9, issue 5 pp. 683-691, Aug. 1999.*

[2] Seongsoo Lee, Jeong-Min Kim, Suo-IK Chae, "New motion estimation algorithm using adaptively quantized low bit-resolution image and its VLSI architecture for MPEG2 video encoding," *IEEE transactions on circuits and systems for video technology, vol. 8, issue 6, pp 734 -744, Oct. 1998.*

[3] Pickering M.R, Arnold J.F, Frater M.R, "An adaptive search length algorithm for block matching motion estimation," *IEEE transactions on circuits and systems for video technology, vol. 7, issue 6, pp 906-912, Dec. 1997.*

[4] Jo. Yew. Tham, Surendra Ranganath, Maitreya, Ashraf Ali Kassim, "A novel unrestricted centre biased diamond search algorithm for block motion estimation," *IEEE transactions on circuits and systems for video technology, vol. 8, issue 4, pp 369-377, Aug. 1998.*

[5] Huan-Sheng Wang, Mersereau R. M, "Fast algorithm for the estimation of motion vectors," *IEEE transactions on image processing, vol. 8, issue 3, pp 435-438, Mar. 1999.*

[6] Jon Wong Kim, Sang UK Lee, "Hierarchical variable block size motion estimation technique for motion sequence coding," optical *engineering, vol. 33, pp. 2553-2561, 1994.*

[7] H.264 AVC: 'Draft ITU-T recommendation and final draft international standard of joint video specification (ITUT Rec. H.264/ISO/IEC14496-10AVC', in *'Joint Video Team (JVT) of ISO/IECMPE Gland ITU-TVCEG', JVT G050, 2003.*

[8] Richardson I.E.G.: '*h.264 and mpeg-4 video compression: video coding for next-generation multimedia'* John Wiley & Sons, 2003.

[9] Tung-Chien Chen, Shao Yi Chien, Yu-Yeh Chen, To-Wei Chen, Liang-Gee Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder", *IEEE TCSVT, v. 16, no. 6, Jun. 2006, pp. 673-688.*

[10] Vanne J, Aho.E, Hamalainen T.D, Kuusilinna. K, "A high-performance sum of absolute difference implementation for motion estimation", *IEEE TCSVT, v. 16, n. 7, Jul. 2006, pp. 876-883.*

[11] Yufei. L, Feng Xiubo, Wang Qin, "A high-performance low cost SAD architecture for video coding". *IEEE TCE, v. 53, n. 2, May 2007. pp. 535-541.*

[12] Liu Zhenyu, Song Yang, Shao Ming, Li Shen, Li Lingfeng, Goto Satoshi, Ikenaga Takeshi, "32-Parallel SAD tree hardwired engine for variable block size motion estimation in HDTV 1080P real-time encoding application", *IEEE SiPS, 2007, pp. 675-680.*

[13] Stephan Wong, et al., "A sum of absolute differences implementation in FPGA hardware", *international journal of electrical and computer engineering 4:9 2009.*

[14] C Hisham, K Komal, Amith K Mishra, "Low power and less area architecture for integer motion estimation", *international journal of electrical and computer engineering 4:9 2009.*

[15] S. Kappagantula, et al. "Motion compensated predictive coding", *Proceedings of international symposium. SPIE, San Diego, CA, August 1983.*

[16] S. Vassiliadis, E.A Hakkennes, J.S.S.M Wong, G.G. Pechanek, "The sum-absolute-difference motion estimator accelerator", *Proceedings of the 24th Euromicro conference, 2000.*