IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 41, No. 4, August, 2024, pp. 2195-2201
Journal homepage: http://iieta.org/journals/ts

Low-Power Approximate SAD Design for Efficient Integer Motion Estimation in Video Compression

Sunitha Rani Madavaram[1*], Mary Valantina George[2]

[1] Department of ETCE, Sathyabama Institute of Technology, Chennai 600119, India
[2] Department of CSE, Saveetha School of Engineering, (SIMATS), Chennai 602105, India

Corresponding Author Email: blessey.ch@gmail.com

https://doi.org/10.18280/ts.410448

Received: 7 January 2024
Revised: 10 May 2024
Accepted: 15 June 2024
Available online: 31 August 2024

Keywords:
Adders, area, delay, power, video coding, integer motion estimation(IME), sum of absolute difference (SAD), high efficiency video coding (HEVC)

ABSTRACT

Integer motion estimation (IME), which leverages the temporal redundancy in video sequences, plays a crucial role in enabling effective video compression. However, the substantial computational demands of real-time motion estimation necessitate significant processing power. This requirement poses a challenge for lightweight mobile devices, which are constrained by limited processing capabilities and rely heavily on battery power. To address this issue, an efficient design for the Sum of Absolute Differences (SAD) module has been proposed, aimed at reducing both computational complexity and power consumption in video encoding applications. The size, power, and speed of the proposed design have been evaluated and integrated using Xilinx tools. Comparative analysis indicates that the proposed design is more compact and energy-efficient than existing models. Additionally, an image sharpening approach has been employed to assess the video quality metrics, including the peak signal-to-noise ratio (PSNR) and accuracy, of the proposed and existing methods. The findings suggest significant improvements in both the performance and energy efficiency of the motion estimation process.

1. INTRODUCTION

High Efficiency Video Coding (HEVC) is a technique that transfers data without loss and compresses it with high efficiency. The novel standard for video compression called HEVC has double the coding efficiency when compared with the standard H.264/AVC [1, 2]. Video encoders are becoming more and more efficient because of their sophisticated digital processing algorithms. These algorithms are employed in specialized hardware to get the best possible balance between power consumption and quality. However, the hardware approach consumes more energy overall. As the mobility of video processing applications increases, power efficiency is becoming a significant concern [3]. Generally, HEVC encoding and decoding will consume more power and have a complex structure due to their encoding and decoding processes. As the algorithm consumes more power, the battery life span will be shorter. As the battery life span is shorter, the sensor systems will lose more data. To overcome these drawbacks, several researchers have researched this area to reduce the area and power consumption in HEVC. Several video coding techniques have been developed to reduce the errors present in the system, which is known to be somewhat error-tolerant when it comes to how human sensory systems interpret the finished output [4]. Consequently, systems that employ video coding can become more energy-efficient through the use of approximation computing [5]. As such, power is significantly increased when approximation computing is integrated into hardware video encoders [6]. The integer motion estimation (IME) module of a video encoder is the most crucial and power-hungry module.

IME is the one eminent power-hungry architecture in video encoder and decoder designs. Several researchers have put their effect on the reduction of area for the IME, which is used in sum of absolute differences (SAD) designs. Nevertheless, these architectures frequently require heavy energy consumption. But these architectures generally use a lot of electricity. So the main challenge of this research is to implement the motion estimation block by minimizing hardware complexity in the SAD architecture. Zhu et al. proposed the truncation-error-tolerant adder and its application in digital signal processing, which has less area and power. The authors Porto et al. proposed motion estimation using approximate arithmetic, which is an efficient hardware design [7]. Next, Rafael Ferreira described the approximate subtractor for video coding for hardware accelerators. The authors [8] explained the usefulness of video coding hardware accelerators by using an approximate subtractor, by which the power consumption has been reduced. The approximate subtractor also plays a key role in the division operation [9-11].

The rest of the organization of the paper is as follows: The overview of the method is described in Section 2. While Section 3 describes the current procedures, Section 4 provides the designed work. Section 5 describes the results of the evaluation of the proposed work. The paper ultimately concludes in Section 6.

2. OVERVIEW

2.1 SAD

The SAD distortion metric is frequently used in IME to choose candidate blocks that complement the best block in the present unit [12, 13]. The SAD formula is as follows:

$$\mathrm{SAD} = \sum_{l=0}^{n-1} \sum_{k=0}^{m-1} |C_{l,k} - R_{l,k}| \quad (1)$$

The SAD analysis is given by Eq. (1), where C=current unit, R= reference unit.

Samples with dimensions denoted by *l* & *m*, by using Eq. (1) we can calculate the addition of two numbers.

3. EXACT COMPRESSOR ADDERS

We need 2 adders to design the exact compressor adder. Generally, these adders require 3 o/p and 5 sample inputs for calculating the sum and carry. Since compressor C_{out} is independent of C_{in}, the structure may be built more efficiently [14, 15]. The overall result (T42) for a precise 4:2 compressor is given by Eq. (2).

$$T_{4:2} = 2(Carry + C_{out}) + Sum \quad (2)$$

Three perfect 4:2 compressors combine to form the precise 8:2 compressor, as shown in Figure 1. Six carry outputs (C_{out4}, Cout3, Cout2, Cout1, Cout0), eight primary inputs, and five carry outputs (C_{out0},C_{out1},C_{out2},C_{out3},C_{out4}) make up the 8:2 precise compressor. Eq. (3) displays the sum output for the precise 8:2 compressor adder [16].

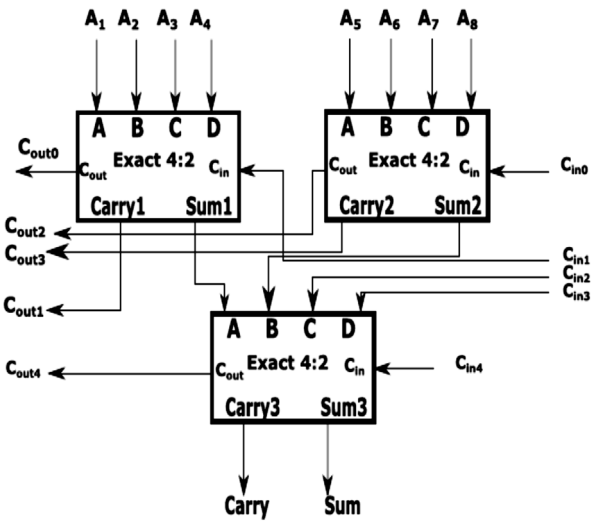

Figure 1. 8:2 compressor adder

3.1 N-Bit exact compressor adder

One 8:2 compressor is all that is required to add 8 inputs with a bit width of 1. There are additional 8:2 compressors when the bit widths of the input operands are increased, and these compressors must be combined. The N-bit precise compressor adder is shown in Figure 2.

$$T_{8:2} = 2(Carry + C_{out4} + C_{out3} + C_{out2} + C_{out1} + C_{out0}) + Sum \quad (3)$$

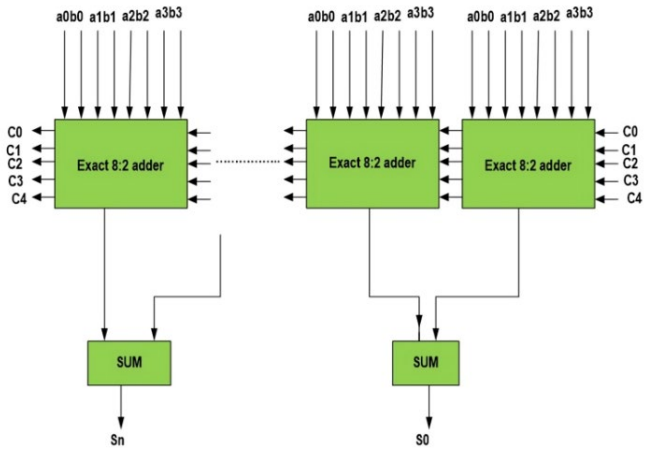

Figure 2. 8:2 compressor adder

3.2 Approximate compressor adder

In the approximation, compressor adders are discussed. Approximately four out of every five compressors have two outputs and four inputs. The sum is in full (T′). Eq. (4) gives the approximate compressor's 4:2 results.

$$T'_{4:2} = 2(Carry) + Sum \quad (4)$$

A large number of compressor adder approximations are known from the literature. We'll discuss a few of them in this paragraph right now is shown in Figure 3. The research [10] suggested using logic gates such as AND, XOR, OR, and compressors. Out of sixteen, the compressor produces an error five times. The efficiency of the compressor decreases as area and power increase. The research [11] provided an example of how to create sum and carry outputs using XOR, AND & OR logic gates. The compressor adder generates an error 4 times out of every 16. Despite achieving impressive precision, this compressor requires additional gates in order to provide sum & carry, which results in issues with area, latency, and power. In order to generate the carry with fewer gates, Minho rebuilt the compressor adder. The compressor adder uses NOR, XNOR, and OR logic gates to generates carry & sum o/ps [12]. The compressor adder generates an error. The compressor requires less room, time, and energy and operates with exceptional precision.

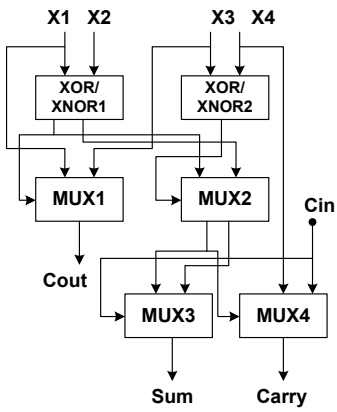

Figure 3. Types of approximate compressor adders

3.3 The mathematical calculations of SAD

One of the IME parts of contemporary video encoders that consume power is the SAD architecture. As a result, SAD unit tuning has been the focus of most current research [6]. Recommends a three-stage SAD architecture based on the minimal SAD determination units, the compression unit, and the absolute difference.

The low-power SAD architectures have been explained by Bossen [7] on the tool's adder, utilizing 3:2 and 4:2 compressor adder trees to produce SAD architecture. A SAD architecture was used by the authors, which comprises reassembling smaller SAD components into larger blocks. The authors offered a method for analyzing the reduction technique, which reduces the number of SAD computations made during IME by removing certain pixels from the JM and x264 software. The author offers a compressor adder based SAD design that is power-efficient by using two 8:2 adder compressors and several adders in the recombination line. Several authors developed an effective 8:2 compressor adder-based SAD design using four different 8:2 compressor adders. Using 8:2 compressors, five separate adders were used to create the recombination line for the 8-bit adder.

The research [8] gives a brief idea of how to design the SAD using the carry cutback approximate adder with fixed-point implementation and floating-point precision. In this, more errors will be present. In paper [12], the approximate compressor adder is used in the Vedic multiplier and the same concept is applied in SAD.

3.4 Inaccurate mathematical procedures in SAD

The authors integrated approximation adders into the SAD design to decrease power consumption. A less power consumption SAD architecture using approximation adders is shown in the research [5] to have a little influence on coding efficiency. The trade-off analysis was carried out by the authors [6] with a single-adder setup. The coding efficiency against power consumption for LOA with five approximation bits uniformly applied to the adders in the tree is displayed in the results. Three distinct approximation adders were used in three separate SAD design positions by Bossen [7]. A C++model was developed to specify the SAD architecture, based on the SAD accelerator that was made available in the research [8]. To reduce the power consumption of HEVC, the study [3] developed an approximate architecture for motion estimation. The authors of the study [9] provided a basic subtractor for SAD and looked at the various architectures made available for HEVC motion estimation. After assessing all approximation adders—including their own—in terms of HEVC motion estimation, the authors published an approximation adder in the study [10].

4. PROPOSED SAD ARCHITECTURE BASED ON APPROXIMATE COMPRESSOR ADDER

Figure 4 shows the SAD design with an adder tree. No advantage of the values on the adder tree is used in the computation. Selvo et al. [11] replaced the adder tree with the same 8:2 compressor that is depicted in Figure 5. An N-bit approximate 8:2 compressor was used in this study rather than the N-bit exact 8:2 compressor adder to save power usage. Only in the case where there are N approximate 8:2 compressors can an N-bit approximate 8:2 compressor be generated.

4.1 Proposed 4:2 compressor adder

The exact 8:2 compressor has two primary outputs, five carry outputs, eight primary inputs, and five carry outputs. To save power, we proposed an approximately 8:2 compressor, which reduces calculation complexity. The main objective of this work is to create an approximate 8:2 compressor. The roughly 8:2 compressor that is being proposed consists of eight inputs (A1, A2, A3, A4, A5, A6, A7, and A8) and two carry inputs (cin0 and cin1). This compressor adder features two carry outputs (Carry1 and Carry2) in addition to its two major outputs (Sum and Carry). The approximate construction of an 8:2 compressor is shown in Figure 5. As Figure 5 shows, three about 4:2 compressors are used to implement.

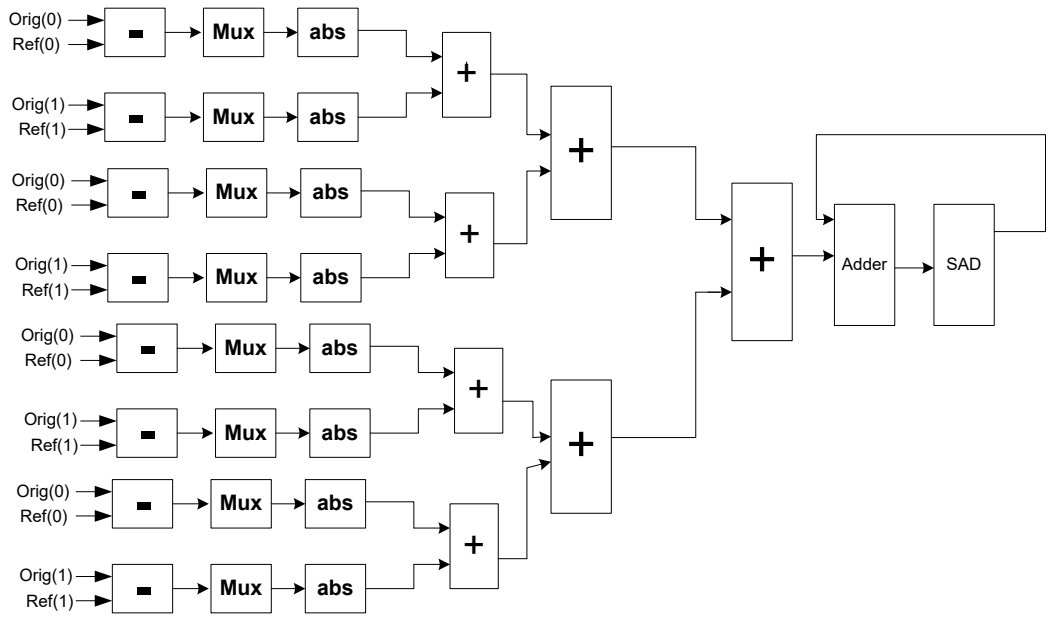

Figure 4. SAD design

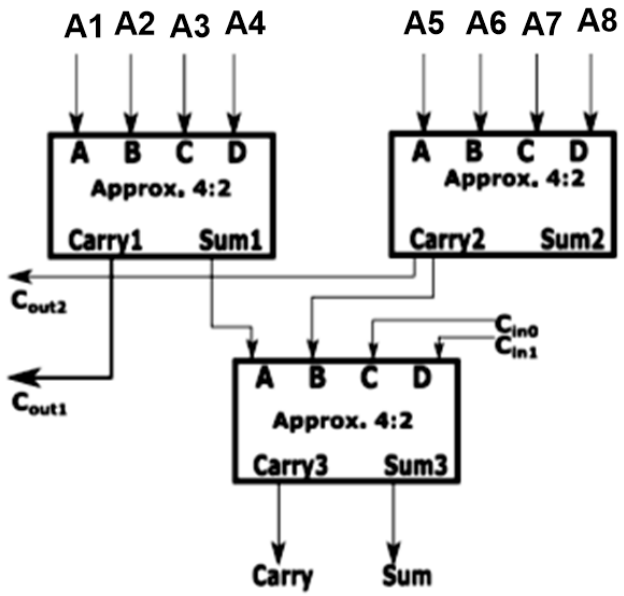

Figure 5. Approximate 8:2 compressor adder

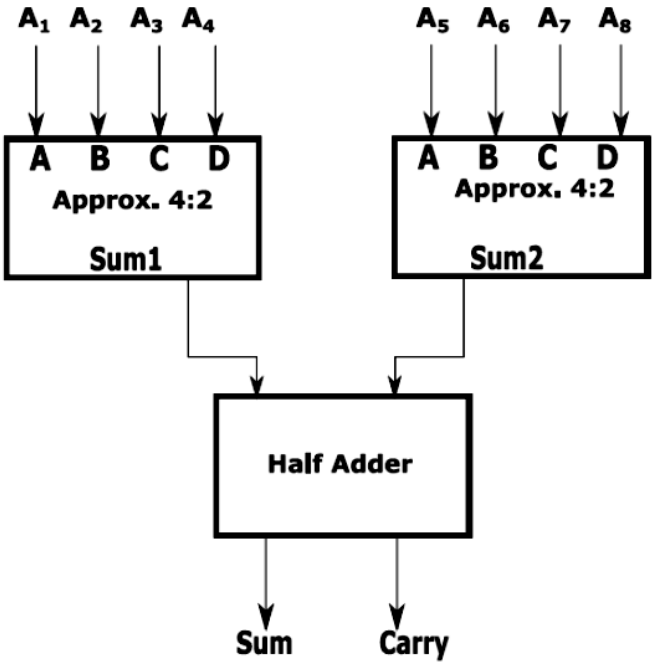

Figure 6. Novel compressor adder

$$T'_{8:2} = 2(Carry + carry2 + carry1) + Sum \qquad (5)$$

The approximate 8:2 compressor adder has three carry outputs rather than six, as shown by Eqs. (3) and (5). In Section 2.3, a number of the approximate 4:2 compressor adder designs currently in use were thoroughly investigated. We suggested six additional 8:2 approximation compressor adders using these roughly 4:2 compressor adders. Out of the six approximately 4:2 compressor adders, as shown in Figure 6. Only UCAC1 has one output, while the other five have two. Because of this, the approximately 8:2 compressor adder constructed with UCAC1 is not built like the previous 8:2 compressor adders, which were built with about 4:2 compressor adders. The architecture of the remaining five approximate 8:2 compressor adders is depicted in Figure 5, and the architecture of the approximate 8:2 compressor adder employing UCAC1 is shown in Figure 6.

4.2 Proposed NXN approximate compressor adder

The proposed NXN approximate compressor adder is implemented using 8 inputs with a size of N bits of the 8:2 compressor adders. The basic compressor adder circuit is shown in Figure 7. With the help of that, we enhanced the N*N proposed compressor adder, which is shown in Figure 7. It contains elements from j=1 to N. When we observe Figure 7, it has two types of regions. One is an exact region and other one is an approximate region. The exact region is made up of a base adder circuit, and the second region is designed using the proposed 8:2 compressor adder. There is an alternative to the exact region and the approximate region in the N*N adder. By having this concept, we can minimize the power, as shown in Figure 8. The carry outputs are passed to the exact adders to get the final C_{out} values.

Since the approximation compressor only has two carry outputs and the exact compressor adders require five inputs, we added three carry sources with a value of zero while shifting from the approximate region to the correct region. Using this method, we obtain the final sum and proceed to process the outputs.

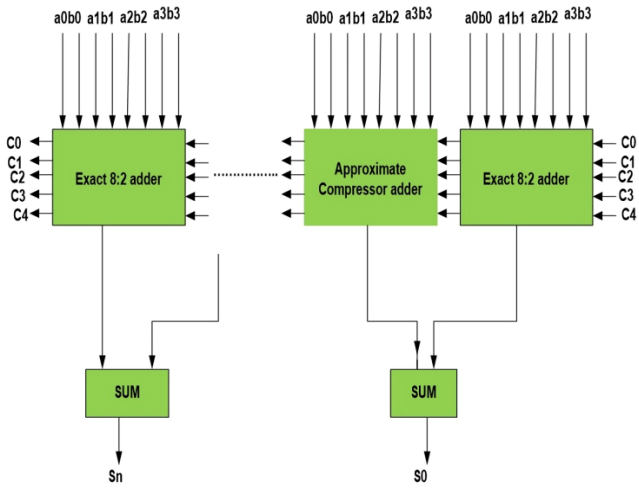

Figure 7. Proposed approximate adder

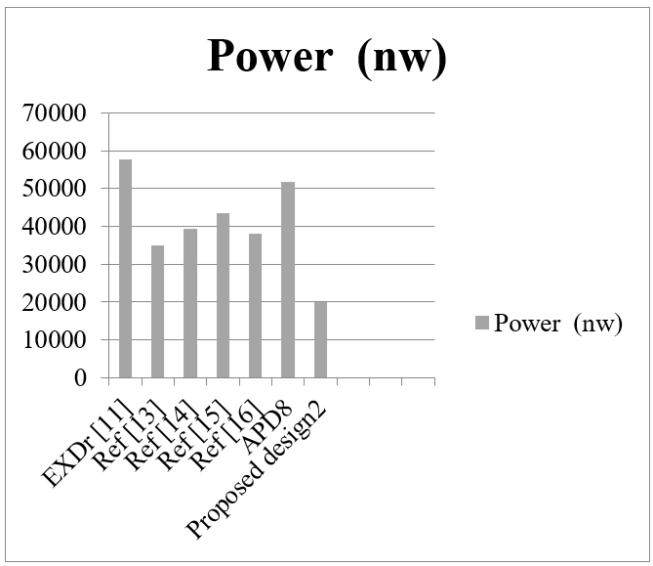

Figure 8. Power consumption report

5. RESULT ANALYSIS

The proposed design has been implemented on a Spartan-6 FPGA, specifically the XC6SLX16-2 device, using the Xilinx tool for area, power, and delay calculations. The proposed design operates under a voltage condition of 1.88V at 26°C with a frequency of 1GHz. The design has been written in Verilog HDL, and the value change dump (VCD) file has been generated to determine the switching activity of the design. The SAD values are derived from the HEVC encoder. The number of errors produced by the proposed approximate compressor adder is significantly lower compared to existing designs. The area occupied by the proposed design is 30% less compared to the architectures in researches [12, 13], as shown in Figure 9. Additionally, the power consumption of the design is lower than that of all existing compressed adder designs, as shown in Table 1 and illustrated in Figure 10.

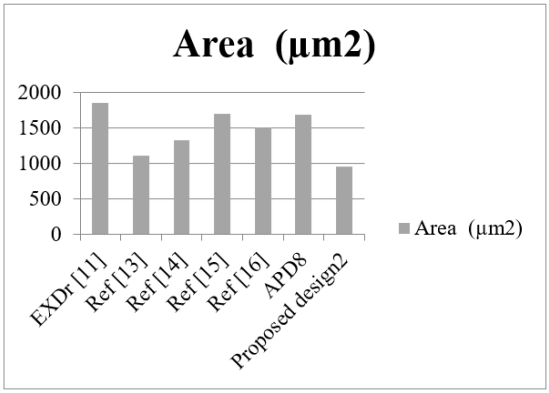

Figure 9. Area report

(a) Real image (b) Existing image [11] (c) Proposed design image [12]

Figure 10. (a) Original image (b) Existing image (c) Proposed design image

Synthesis results of existing and proposed designs using parameters power, delay, area, and number of errors are shown in Table 1.

Table 1. Synthesis results

SAD	Power (nw)	Delay (ps)	Area (μm^2)	No. of Errors
45nm technology				
EXDr [11]	57847	70.06	1850	0
Existing design ref. [12]				
Ref. [13]	34912	70.00	1117	10
Ref. [14]	39333	61.29	1334	10
Ref. [15]	43489	61.00	1698	10
Ref. [16]	38156	64.11	1512	10
Proposed design				
APD8	51887	69.95	1691	10
Proposed design2	19875	34.58	956	40

Table 2. Synthesis result of proposed and existing design using Xilinx-Sparten6 FPGA kit

S.No	Design	No. of Slices	LUTs	Registers
1	Existing design [11]	234	50	17
2	Existing design [16]	146	39	14
3	Proposed design	101	22	11

The Xilinx-Sparten6 implementation using FPGA kit. The proposed design has fewer slices and look up tables when compared with the existing designs [17]. The number of registers required is also less, as shown in above Table 2.

The PSNR ratios for the existing and proposed designs are displayed in Table 3. The corresponding graph is displayed in Figure 11.

Table 3. PSNR ratio for proposed and existing designs

S.No	Methods	PSNR(db)
1	Real Image	29.457
2	Existing Method	27.28
3	Proposed Design	21.38

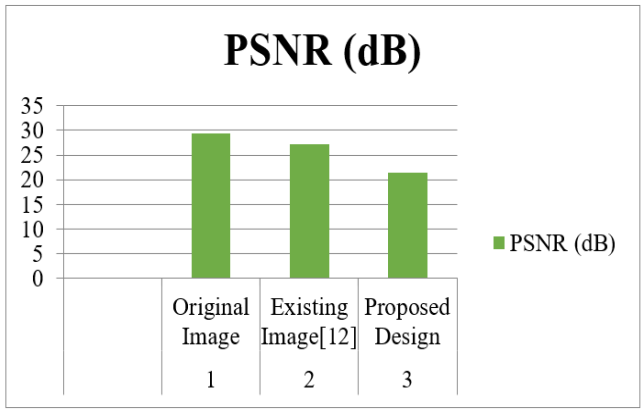

Figure 11. PSNR graph

Application of Image Processing

The proposed SAD-based compressor adder [18, 19] has been implemented in a grayscale image of 512*512 image sharpening. An image has been chosen for analyzing the quality of the proposed SAD-based compressor adder [20]. The proposed image has less error when compared with existing methods, and the PSNR for the proposed image has been calculated and has a lower PSNR ratio when compared with existing method, which has high accuracy in the proposed image. Table 4 shows the results of the accuracy, PSNR, and quality metrics of the proposed and existing methods. Figure 10 shows the output image of size 512×512 (a) Original image (b) Exact multiplier and (c) the proposed SAD-based compressed adder design.

The accuracy rate of the SAD-based compressor adder design is displayed in Table 4 and Figure 12 gives its graph.

Table 4. Accuracy of proposed and existing images

S.No	Methods	Accuracy (%)
1	Original Image	96.01
2	Existing Method [12]	97.11
3	Proposed Image	99.02

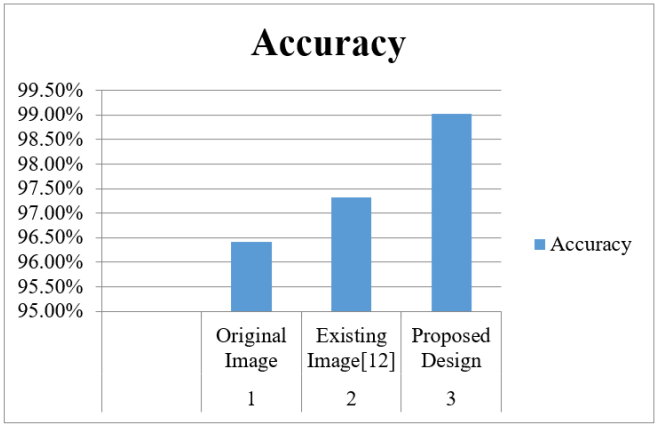

Figure 12. Accuracy graph

6. CONCLUSION

The proposed SAD design is primarily used in image processing applications where the approximation concept has been applied. The approximate computing concept reduces the area, power, and speed parameters. By using the approximate computing concept with minimal error, approximately 75% of the area of the existing architecture is reduced. The proposed design has been tested in the HEVC encoder, demonstrating efficient area and power utilization. The HEVC encoder also exhibits high-speed performance when compared with existing designs. The paper likely proposes techniques for approximating the SAD calculation, a core operation in IME, using less power-hungry hardware implementations. Low-power approximate SAD designs have the potential to significantly improve the power efficiency of video compression algorithms. This is particularly relevant for battery-powered devices like smart-phones and wearable cameras. By achieving a balance between power savings and acceptable video quality, such techniques can enable longer battery life and enhance user experience for video applications on these devices. Research could also investigate the integration of these low-power SAD designs with advanced motion estimation algorithms for optimal performance. Additionally, exploring hardware architectures specifically optimized for implementing approximate SAD calculations with minimal power consumption could be a promising avenue. Low-power approximate SAD design offers a compelling approach to reducing the power footprint of video compression. By delving deeper into this area and addressing the trade-off between power and quality, significant advancements can be made in energy-efficient video processing for portable devices.

ACKNOWLEDGEMENT

This work is supported by Research & Development Lab in Satyabhama Institute of Science and Technology, Chennai and also supported by Vidya Jyothi Institute of Technology, Hyderabad.

REFERENCES

[1] Dutt, S., Nandi, S., Trivedi, G. (2016). A comparative survey of approximate adders. In 2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA), Kosice, Slovakia, pp.61-65. https://doi.org/10.1109/RADIOELEK.2016.7477392

[2] Kahng, A.B., Kang, S. (2012). Accuracy-configurable adder for approximate arithmetic designs. In Proceedings of the 49th Annual Design Automation Conference, San Francisco, USA, pp. 820-825. https://doi.org/10.1145/2228360.2228509

[3] Zhu, N., Goh, W.L., Zhang, W., Yeo, K.S., Kong, Z.H. (2009). Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 18(8): 1225-1229. https://doi.org/10.1109/TVLSI.2009.2020591

[4] Naga Jyothi, G., Sri Devi, S. (2017). Distributed arithmetic architectures for fir filters-a comparative review. In 2017 International Conference on Wireless Communications, Signal Processing and Networking (Wisp NET), Chennai, India, pp. 2684-2690. https://doi.org/10.1109/WiSPNET.2017.8300250

[5] Shafique, M., Ahmad, W., Hafiz, R., Henkel, J. (2015). A low latency generic accuracy configurable adder. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, USA, pp. 1-6. https://doi.org/10.1145/2744769.2744778

[6] Nagajyothi, G., Kumar, G.P., Kumar, B.S., Kumar, B.D., Damodaram, A.K. (2023). High-Speed low area 2D FIR filter using Vedic multiplier. In Proceedings of Third International Conference on Advances in Computer Engineering and Communication Systems: ICACECS 2022, Singapore: Springer Nature Singapore, pp. 569-578. https://doi.org/10.1007/978-981-19-9228-5_48

[7] Bossen, F. (2013). Common test conditions and software reference configurations. JCTVC-L1100, 12(7): 1.

[8] Porto, R., Agostini, L., Zatt, B., Porto, M., Roma, N., Sousa, L. (2017). Energy-efficient motion estimation with approximate arithmetic. In 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP), Luton, UK, pp. 1-6. https://doi.org/10.1109/MMSP.2017.8122248

[9] Naga Jyothi, G., Sriadibhatla, S. (2019). ASIC implementation of low power, area efficient adaptive fir filter using pipelined DA. In Microelectronics, Electromagnetics and Telecommunications: Proceedings of the Fourth ICMEET 2018, Springer Singapore, pp. 385-394. https://doi.org/10.1007/978-981-13-1906-8_40

[10] Paltrinieri, A., Peloso, R., Masera, G., Shafique, M., Martina, M. (2018). Approximate-computing architectures for motion estimation in HEVC. In 2018 New Generation of CAS (NGCAS), Valletta, Malta, pp. 190-193. https://doi.org/10.1109/NGCAS.2018.8572080

[11] Selvo, P., Masera, M., Peloso, R., Masera, G., Shafique, M., Martina, M. (2019). An optimized partial-distortion-elimination based sum-of-absolute-differences architecture for high-efficiency-video-coding. In Applications in Electronics Pervading Industry, Environment and Society: APPLEPIES 2018 6. Springer International Publishing. Springer, Cham, pp. 245-251. https://doi.org/10.1007/978-3-030-11973-7_28

[12] Ferreira, R., Leme, M., Corrêa, M., Agostini, L., Diniz, C., Zatt, B. (2019). Approximate subtractor operator for low-power video coding hardware accelerators. In 2019

26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Genoa, Italy, pp. 426-429. https://doi.org/10.1109/ICECS46596.2019.8964783

[13] Ahmad, W., Ayrancioglu, B., Hamzaoglu, I. (2020). Comparison of approximate circuits for H. 264 and HEVC motion estimation. In 2020 23rd Euromicro Conference on Digital System Design (DSD), Kranj, Slovenia, pp. 167-173. https://doi.org/10.1109/DSD51259.2020.00036

[14] Tasoulas, Z.G., Zervakis, G., Anagnostopoulos, I., Amrouch, H., Henkel, J. (2020). Weight-oriented approximation for energy-efficient neural network inference accelerators. IEEE Transactions on Circuits and Systems I: Regular Papers, 67(12): 4670-4683. https://doi.org/10.1109/TCSI.2020.3019460

[15] Ansari, M.S., Mrazek, V., Cockburn, B.F., Sekanina, L., Vasicek, Z., Han, J. (2019). Improving the accuracy and hardware efficiency of neural networks using approximate multipliers. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 28(2): 317-328. https://doi.org/10.1109/TVLSI.2019.2940943

[16] Burger, W., Burge, M.J., Burge, M.J., Burge, M.J. (2009). Principles of digital image processing. London: Springer, Vol. 111. https://doi.org/10.1007/978-1-84882-919-0

[17] Kumar, U.A., Guturu, S., Ahmed, S.E. (2022). Design and exploration of low-power SAD architectures using approximate compressors for integer motion estimation. Microprocessors and Microsystems, 94: 104659. https://doi.org/10.1016/j.micpro.2022.104659

[18] Basha, S.M., Kannan, M. (2017). Design and implementation of low-power motion estimation based on modified full-search block motion estimation. Journal of Computational Science, 21: 327-332. https://doi.org/10.1016/j.jocs.2016.11.018

[19] Vasanthapriya, K., Vaishnavi, P., Nithya Lakshmi, A. (2017). Implementation of sad algorithm for motion estimation. IJARIIE, 3(2): 3078-3085.

[20] Paltrinieri, A., Peloso, R., Masera, G., Shafique, M., Martina, M. (2019). On the effect of approximate-computing in motion estimation. Journal of Low Power Electronics, 15(1): 40-50. https://doi.org/10.1166/jolpe.2019.1592

NOMENCLATURE

MV	Motion Vector
BD	Best Distance
ME	Motion Estimation
BMA	Motion Matching Algorithm
HEVC	High Efficiency Video Coding