Design and Implementation of Energy Efficient Approximate Window based Adder Architecture

B. Anish Fathima[1], Dr.C. Vasanthanayaki[2]
[1]*Sri Krishna College of Engineering and Technology, Coimbatore,TamilNadu, India*
[2]*Government College of Technology, Bodinayakanur,TamilNadu,India*

ABSTRACT: *Many Signal and Image processing applications require complex mathematical models and computations. Rigorous number of Additions and Multiplications are performed on an average to perform convolution and filtering applications and leads to power consumption. Fixed point and floating-point arithmetic computations are finding applications in Machine learning applications like classification, segmentation and analysis. Hence energy and power consumptions of these architectures are the major challenges faced today. Adders are the basic building block of many filtering applications. Approximate Adders proposed will help the entire architecture to have a lesser energy and power consumption.*

KEYWORDS *–Adder Architecture, Approximation, Energy efficiency, Segmentation*

I. INTRODUCTION

The rapid emergence of computing devices especially wearables and smart devices including real time processing requires the efficient number of computations and additions play a very vital role. Different adders have emerged for over past four decades. Researchers find possible solutions to bring out an optimized approximation technique for fixed point addition arithmetic operations.

The Adders have been broadly classified based on their methodology of working as inexact adders based on voltage scaling, inexact adders based on segmentation, Inexact adders based on changes in Transistor level schematic. Research at various levels of abstraction from system level to transistor level happening in the Approximation techniques. Here the interest is on system level. The challenges faced are encountered with parameters like Area overhead utilized, power consumption, propagation delay incurred and hence the overall energy conservation [11].

The Adder that is being approached here is segmentation-based adders. The Adder operands are segmented and approximated by Least Significant rejection category. Such adders will be very helpful when the computation operand is wide.

As the Segmentation done is very close to the original operand, the approximate product is very much closer to the original one. The error percentage can be reduced as much as less than 1%.

The rest of this article is categorized as follows. Section II details the evolution of various approximate addition techniques. Section III renders a detailed description about the proposed approximate addition technique. Section IV discusses simulation and implementation results. Finally, Section V concludes this article.

II. APPROXIMATION TECHNIQUES

An emerging concept in design called approximate computing takes advantage of applications' innate tolerance to computational inaccuracies to enable extremely efficient hardware and software implementations. Promising findings have been obtained from a number of earlier attempts that investigated approximation computing in hardware and software. Hardware techniques alter the architecture at different levels of abstraction to introduce choices between output quality and efficiency, while software techniques usually enhance

performance by avoiding computations or by using less expensive operations like inter-thread synchronization. Approximate computing has proven to have great potential, as demonstrated by these works. Several error criteria, including error magnitude, relative error, average error magnitude, and error probability (error rate), can be used to assess the quality of a certain n-bit approximation arithmetic circuit. [2].

For error-tolerant applications that can accept a small amount of accuracy loss in exchange for better performance and energy efficiency, approximate circuits have been investigated. Multimedia, recognition, and data mining applications are examples of applications that are naturally error-tolerant and do not necessitate flawless computational precision. Approximate circuits could be a viable substitute for digital systems that can withstand some accuracy loss in these applications, helping them perform better in terms of energy efficiency by lowering area, power, and latency. These have the intriguing characteristic of having a high level of internal algorithmic robustness to extrinsic and intrinsic "errors." Large volumes of input data with high redundancy and the potential for significant defects or noise are processed by these algorithms [4].The algorithms typically use iterative, successive refinement techniques, which imparts them with a self-healing nature since subsequent iterations may correct errors introduced in previous iterations.

Earlier research has investigated a number of methods that take advantage of computational error tolerance to increase the energy efficiency of multipliers. Three categories can be used to group them: Unreliable building blocks are used; bit-width truncation occurs; and aggressive voltage scaling occurs. A novel systematic method for designing effective hardware implementations for algorithms that exhibit intrinsic error resilience was introduced in [9], and it was based on the idea of scaled effort hardware. If exact outputs are not needed for the process, truncation techniques can significantly increase the efficiency of digital addition.

To improve energy efficiency of multipliers, previous studies have explored various techniques exploiting computational error tolerance. They can be classified into three categories: Aggressive voltage scaling; Truncation of bit-width; Use of inaccurate building blocks. In [9], a new systematic approach based on the concept of scalable effort hardware, for the design of efficient hardware implementations for algorithms that demonstrate inherent error resilience was presented. The efficiency of digital addition can be improved tremendously by truncation methods provided precise outputs are not required for the operation. A new truncation scheme based on multiplexers was proposed in the paper [8] that has a lower average and mean square error than the existing methods. Trading of accuracy for power can be done with the under-designed multiplier architecture presented in [7], which has a mean error of 1.39% - 3.35% and power savings between 30% - 50%. The design of a low-power multiplier using an energy-efficient full adder was proposed [6].
The paper [5] reviews recent advances in approximate computing, focusing on approximate circuit design, associated error metrics, and algorithm-level techniques.

III. PROPOSED APPROXIMATE ADDITION TECHNIQUE

3.1 Segment Based Addition

Multiplying energy efficiency is an important goal. Many digital applications use fixed-point arithmetic, which allows for computational errors. This project proposes a multiplier that can balance computational accuracy and energy consumption. "The main principle is to segment the original operand by the significant bits and perform addition only on these segments." The proposed technique is different from existing techniques such as aggressive voltage scaling. Differently, it utilizes segments of important operands to perform approximate addition. Reduce bit width and use imprecise components for approximate addition. The general steps for segmentation-based addition are shown below.
The general process for addition using the segmentation method is as follows if two n-bit operands need to be multiplied:

With two n-bit operands (let's say 16-bit operands),

Each n-bit operand has an m-bit (let's say 8-bit) segment that needs to contain the leading one bit. These two m-bit segments are multiplied, and the 2m-bit product is expanded to the 2n-bit product.

3.2 Techniques

Three techniques are used depending on how a segment is chosen from the operand:

- Dynamic Segment Method (DSM)
- Static Segment Method (SSM)
- Enhanced Static Segment Method (ESSM)

DSM is an older segmentation technique still in use. SSM and ESSM are included in the approximate addition method that is being suggested. The detailed explanation of each technique is provided below.

3.2.1Dynamic Segment Method (DSM)

An m-bit segment is defined as m contiguous bits beginning with the leading bit in an n-bit positive operand. This definition serves to both justify and explain the proposed adder [1]. In contrast to the static segment method (SSM), which will be covered later in this section, we refer to this approach as the dynamic segment method (DSM). When we have two m-bit segments from two n-bit operands, we can use a m ×m adder to multiply. Even with an 8 × 8 adder, we can accomplish 99.4% accuracy with this method for a 16 × 16 addition. As seen in Figure 1, this technique can capture m-bit segments beginning at the precise leading one-bit position.

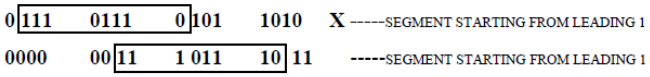

Fig. 1 A sample of Dynamic Segment Method

Because it can eliminate redundant bits (i.e., sign-extension bits) while feeding the most useful m significant bits to the adder, such a addition approach has little effect on computational accuracy. Furthermore, because adders' complexity (and energy consumption) increases quadratically with n, a m×m adder uses a lot less energy than an n×n adder. For instance, the average energy consumption of the 4 × 4 and 8 × 8 adders is approximately 20 × and 5 × lower than that of a 16 × 16 adder per operation.The energy savings from using the m × m adder are entirely negated by the significant area and energy penalties associated with hardware requirements; Because the leading one bit can be anywhere, the area and energy penalties associated with three requirements in DSM are to capture an m-bit segment starting from an arbitrary bit position in an n-bit operand. Thus, it is suggested that the number of starting bit positions that can be used in SSM to extract an m-bit segment from an n-bit operand be limited to two or three at most.

3.2.2 Static Segment Method (SSM)

The Static Segment Method is the suggested technique for approximating addition (SSM). We can take two m-bit segments from two n-bit operands for a addition using the m-bit SSM in four different ways, regardless of m and n. The m-bit segment containing the leading one bit of each operand is selected for a addition, and the selected segments from both operands are applied to the m×m adder. Two design architectures using two different numbers of bits for the segment is given below:

- SSM_8X8_Add – here the window size (m) is 8 bit

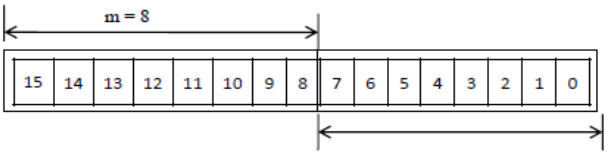

Fig. 2 Windowing in SSM where m=8

- SSM_10X10_Add – here the window size (m) is 10 bit

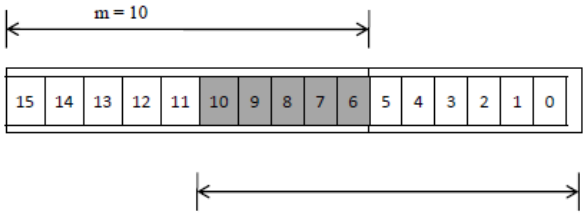

Fig. 3 Windowing in SSM where m=10

By substituting two n-bit LODs and shifters for the DSM with two (n–m)-input OR gates and m-bit 2-to-1 multiplexers, the SSM significantly reduces the complexity of the circuit that selects m-bit segments and directs them to the m×m adder; if the first (n–m) bits starting from the MSB are all zeros, the lower m-bit segment must contain the leading one. Additionally, the SSM enables us to use a 2n-bit 3-to-1 multiplexer in place of the 2n-bit shifter that was employed for the DSM. Figure 4 illustrates how a 2m-bit result can be expanded to a 2nbit result by left-shifting the 2m-bit result by one of three possible shift amounts, since each operand's segment is taken from one of two possible segments in an n-bit operand.

1) There is no shift if both segments come from the lower m-bit segments;
2) There is a shift of (n–m) if two segments come from the upper and lower respective segments;
3) When both segments originate from the upper ones, there is a 2× (n–m) shift.

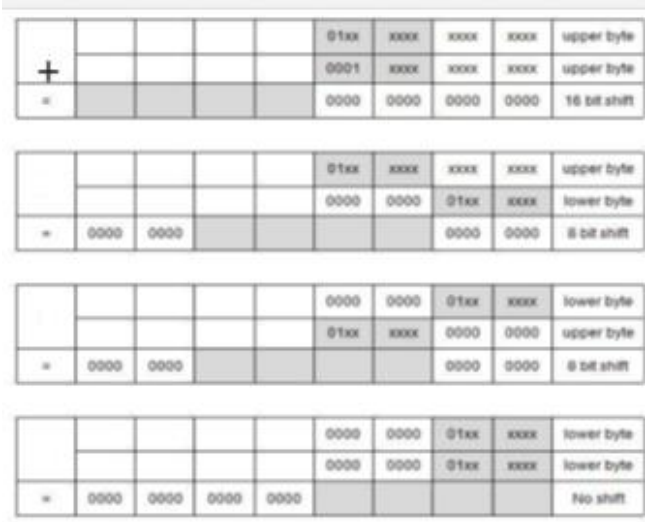

Fig. 4 Various possible shifting processes in SSM

An SSM that allows an m-bit segment to be taken from one of two potential bit positions of an n-bit operand is depicted in Figure 5. Because the complexity (i.e., area and energy consumption) of auxiliary circuits for selecting/steering m-bit segments and expanding a 2m-bit result to a 2nbit result scales linearly with m, the main benefit is that it can be scaled for different values of m and n.

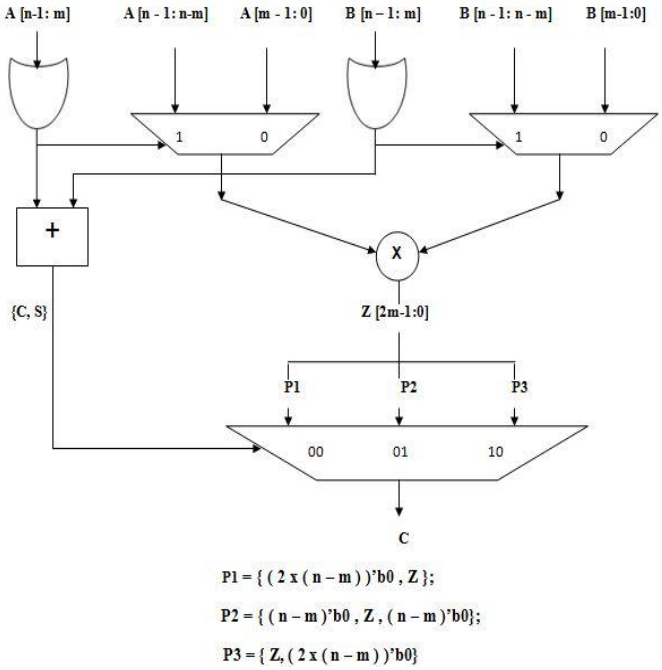

Fig. 5 SSM Block Diagram

We suggest precompute the bit-wise OR value of B[n–1:m] and preselect between the two possible m-bit segments (i.e., B[n–1:n–m] and B[m–1:0]) in Figure, and store them in memory in place of the native B value for applications where one of the operands of each addition is frequently a fixed coefficient.

An SSM with m = n/2 may have appreciably poor accuracy for the operands displayed in the following figure:

+					0000	0001	xxxx	xxxx
					0000	0010	xxxx	xxxx

Fig. 6 A sample for low accurate operands

Here, zeros fill a large number of MSBs of m-bit segments that contain the leading one bit. However, when m increases beyond n/2, the issue becomes less serious; for example, for m = 10, there is overlap in the bits covered by both potential m-bit segments.

3.2.3Enhanced Static Segment Method (ESSM)

The two 2-to-1 multiplexers at the input stage and one 3-to-1 multiplexer at the output stage are replaced with 3-to-1 and 5-to-1 multiplexers, respectively, along with a few small modifications to the logic functions generating multiplexer control signals, in order to support three possible starting bit positions for selecting an m-bit segment where m = n/2. The result of these modifications is the Enhanced Static Segment Method (ESSM), an improved architecture technique

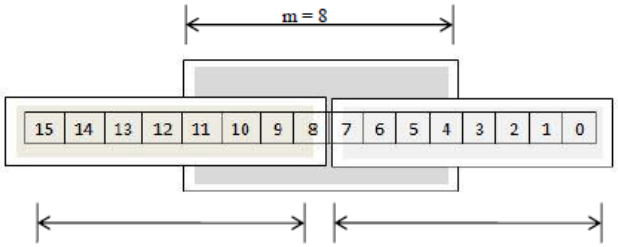

Fig. 7 Windowing in ESSM here m=8

With significantly less energy consumption, this improved SSM design for m = 8 and n = 16 (represented by ESSM8×8) can offer accuracy comparable to that of SSM_10×10.

3.2.4Optimization of Proposed Architecture

Although the suggested ESSM addition method uses less energy than the SSM method, it requires a larger overhead space. Consequently, the SSM algorithm is modified in order to perform additional optimization. This modification allows the SSM method to achieve a significant reduction in area overhead along with improved computational accuracy. The Improved Static Segment Method (ISSM) is the name given to this optimized SSM.

IV. SIMULATION AND IMPLEMENTATION RESULTS

In this project, Verilog Hardware Description Language was used to simulate the design and implementation of energy-efficient adder architectures. These designs' performance metrics were examined using the Xilinx Design ISE (Integrated Software Environment) 13.2. The three performance metrics are area, delay, and power. It was thought that Spartan 6 served as the operating system.

4.1 Evaluation of Approximate Adders in Comparison

Here, the comparison focuses on the approximate adder methods' performance parameters that have been previously discussed. The average computational error (in%) of the product obtained using the various approaches discussed is shown in Table I. For this tabulation, an average of thirty distinct pairs of operands are taken into account. The accuracy of ESSM turns out to be superior.

TABLE I

INVESTIGATION OF AVERAGE CALCULATIVE ERROR

Segment Method	**Avg. computational Error (%)**
SSM_8X8_Add	~5
SSM_10X10_Add	~1
Proposed Methods	
ESSM_8X8_Add	~1
ISSM_8X8_Add	~2

The analysis of performance metrics for each of the four methods, including delay, power, energy, and area consumed, is shown in Table II. While ISSM lowers the area overhead, ESSM turns out to be more energy-efficient.

TABLE II

INVESTIGATION OF DELAY, POWER AND AREA

Segment Methods	**Delay (ns)**	**Power (mW)**	**Energy (pJ)**	**Area (No. of LUTs)**
SSM_8X8_Add	9.53	29	276.37	25
SSM_10X10_Add	10.451	25	261.28	25
Proposed Methods				
ESSM_8X8_Add	10.691	25	**267.28**	49
ISSM_8X8_Add	12.380	31	431.40	**34**

V. CONCLUSION

This brief proposes an approximate adder that balances energy and accuracy. The proposed approximate adder applies these two segments, which include the leading bits from two operands (i.e., SSM), to a m×m adder. It takes m consecutive bits (i.e., an m-bit segment) of an n-bit operand, either beginning at the MSB or ending at the LSB. Compared to a true adder, the suggested Enhanced Static Segment Method uses less power and, consequently, significantly less energy, with an average computational error of ~1%. Additional optimization of the suggested adder is also carried out, resulting in a significant decrease in the proposed adder's area consumption and an improvement in average computational accuracy.

Therefore, in comparison to other approximate adders like the truncated adder, the suggested adder can result in computations for signal and image processing that are of higher quality. The fact that this suggested method maintains quality even in energy-efficient devices is its most significant feature. Better results may also result from keeping one of the addition's operands fixed.

REFERENCES

[1] Srinivasan Narayanamoorthy, Hadi Asghari Moghaddam, Zhenhong Liu, Taejoon Park, and Nam Sung Kim, July 2014, "Energy-Efficient Approximate Addition for Digital Signal Processing and Classification Applications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Early Access Articles, pp. 1 – 5.

[2] M. L. Aranda, R. Báez, and O. G. Diaz, "Hybrid adders for high-speed arithmetic circuits: A comparison," in Proc. 7th IEEE Int. Conf. Elect. Eng. Comput. Sci. Autom. Control (CCE), Tuxtla Gutierrez, NM, USA, Sep. 2010, pp. 546–549.

[3] C. H. Chang, J. M. Gu, and M. Zhang, "A review of 0.18-μm full adder performances for tree structured arithmetic circuits," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 6, pp. 686–695, Jun. 2005.

[4] S. Goel, A. Kumar, and M. A. Bayoumi, "Design of robust, energyefficient full adders for deep-submicrometer design using hybrid-CMOS logic style," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 14, no. 12, pp. 1309–1321, Dec. 2006.

[5] D. Radhakrishnan, "Low-voltage low-power CMOS full adder," IEE Proc.-Circuits Devices Syst., vol. 148, no. 1, pp. 19–24, Feb. 2001.

[6] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits: A Design Perspective, 2nd ed. Delhi, India: Pearson Education, 2003

[7] A. M. Shams, T. K. Darwish, and M. A. Bayoumi, "Performance analysis of low-power 1-bit CMOS full adder cells," IEEE Trans. Very Large Scale Integration. (VLSI) Syst., vol. 10, no. 1, pp. 20–29, Feb. 2002.

[8] C.-K. Tung, Y.-C. Hung, S.-H. Shieh, and G.-S. Huang, "A low-power high-speed hybrid CMOS full adder for embedded system," in Proc. IEEE Conf. Design Diagnostics Electron. Circuits Syst., vol. 13. Apr. 2007, pp.1–4.

[9] M. Vesterbacka, "A 14-transistor CMOS full adder with full voltage swing nodes," in Proc. IEEEWorkshop Signal Process. Syst. (SiPS), Taipei, Taiwan, Oct. 1999, pp. 713–722.

[10] N. H. E. Weste, D. Harris, and A. Banerjee, CMOS VLSI Design: A Circuits and Systems Perspective, 3rd ed. Delhi, India: Pearson Education, 2006.

[11] AnishFathima, B., Mahaboob, M., "Design and Analysis of Energy Efficient Domino Logic Architectures with Single Electron Transistors in Pull Down Network and Keeper Topology", *EAI Endorsed Transactions on Energy Web*, 2021, 8(33), pp. 1–6