Design of Approximate Multiplier for Image Processing

THESIS SUBMITTED

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY IN VLSI DESIGN AND MICROELECTRONICS TECHNOLOGY

BY

SANCHARI GUHA

Class Roll No: 002010703022

Examination Roll No- M6VLS23004

Registration No- 154124 of 2020-2021

Under the guidance of

PROF. JAYDEB BHAUMIK

Department of Electronics and Tele-communication Engineering

JADAVPUR UNIVERSITY, KOLKATA-700032

WEST BENGAL, INDIA

AUGUST-2023

FACULTY OF ENGINEERING AND TECHNOLOGY

JADAVPUR UNIVERSITY

DECLARATION

I hereby submit this thesis entitled "Design of Approximate multiplier for Image processing "for partial fulfillment of *Master of Technology* degree in the stream of *VLSI Design and Microelectronics Technology*. This thesis contains original work and related literature survey.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that as required by these rules and conduct I have fully cited and referenced all materials and results that are not original with this work.

Name: Sanchari Guha

Class Roll No: 002010703022

Examination Roll No: M6VLS23004

Registration No: 154124 of 2020-2021

Project Title: Design of Approximate multiplier for Image processing

Date:

Place: Kolkata

_____

(Student Signature)

FACULTY OF ENGINEERING AND TECHNOLOGY

JADAVPUR UNIVERSITY

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled "**Design of Approximate multiplier for Image processing**" has been submitted by **Ms. Sanchari Guha** (Class Roll No: 002010703022, Examination Roll No: M6VLS23004 and Registration No: 154124 of 2020-2021) of *Jadavpur University* is a record of bonafide research work carried out by her under my guidance and supervision and be accepted in partial fulfillment of the requirement for the degree of *Master* of *Technology in VLSI and Microelectronics Engineering*, in the Department of Electronics and Tele-communication Engineering, *Jadavpur University*, *Kolkata-700032*.

The results presented in this thesis have been verified and are found to be satisfactory.

_____

Prof. Jaydeb Bhaumik

Supervisor

Department of Electronics and

Tele-communication Engineering,

Jadavpur University,

Kolkata- 700032

_____

Prof. Manotosh Biswas

Head of the Department

Department of Electronics and
Tele-communication Engineering,

JadavpurUniversity,

Kolkata-700032

_____

Prof. Ardhendu Ghoshal

Dean

Faculty Council of Engineering
and Technology,

Jadavpur University,

Kolkata-700032

FACULTY OF ENGINEERING AND TECHNOLOGY

JADAVPUR UNIVERSITY

CERTIFICATE OF APPROVAL*

The forgoing thesis, entitled "Design of Approximate multiplier for Image processing" is hereby approved as a creditable study on an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not necessarily endorse or accept every statement made, opinion expressed or conclusion drawn therein but approve the thesis only for the purpose for which it has been submitted.

Committee on Final Examination for
Evaluation of the Thesis:

_____

External Examiner

_____

Prof. Jaydeb Bhaumik
Supervisor
Department of Electronics and
Tele-communication Engineering,
Jadavpur University,
Kolkata- 700032

*Only in case the thesis is approved.

ACKNOWLEDGEMENTS

I express my deep sense of gratitude to my Thesis Supervisor, Prof. Jaydeb Bhaumik, Department of Electronics & Telecommunication Engineering, Jadavpur University, Kolkata, for providing me the opportunity to carry out the Thesis work. I am grateful to him for the valuable insights and suggestions that he gave me throughout my M.Tech. Thesis.

I also thank, Prof. Manotosh Biswas, HOD, Department of Electronics and Telecommunication Engineering.

I also like to express my sincere gratitude to Ph.D. Research Scholars Jhilam Jana and Sayan Tripathi for providing all the necessary help, encouragement, and support in this thesis work.

I would like to express my sincere gratitude to all the faculty members of ETCE Department, Jadavpur University and all other teaching and non-teaching staff of the department for providing necessary support.

Last but not the least, this work would not have been possible without the love, support and encouragement of my family members, classmates, and all of them.

Sanchari Guha

M.Tech. VLSI Design and microelectronics Technology

Jadavpur University

August-2023

ABSTRACT

Today for image processing in smart gadgets, machine learning plays an important role. Low power consumption, fast processing and portability are the essential requirements for designing such smart gadgets. Approximate computing arithmetic circuits are widely employed to satisfy above requirements.

In this thesis efficient design and implementation of approximate unsigned number multiplication unit has been presented. Multiplier is widely used to compute different transformation, convolution, etc.

Proposed and existing designs are analyzed with respect to area, delay and power consumption. In Digital Signal Processing(DSP) several approaches are made to improve present designs.

The modified Dadda Multiplier with different dimensions have been proposed in this thesis, along with their design and analysis, with the goal of reducing output error while applying an accurate compressor. The effectiveness of the modified multipliers is calculated by various experimental assessments of the proposed designs and its parameters are compared with those of the most advanced approximate multipliers. As a result of the findings, it can be seen that the modified multipliers significantly reduce the error rate when compared to the existing approximate multipliers that have been introduced in the literature. The area and the power consumption are both reduced by using the modified multiplier. Furthermore, the latency has been improved in comparison to the existing approximate multiplier. The modified multipliers of 8, 16, and 32 bits are designed and implemented.

CONTENTS

List of Figures:

CHAPTER 1

INTRODUCTION

INTRODUCTION

In recent years, the need for high-performance, energy-efficient systems has grown exponentially in the computing technology. Traditional computer methods frequently incur higher power and delay costs in order to achieve precise and accurate results. In order to overcome these drawbacks, approximate computing can be employed to trade off the precision for improving performance. In order to utilize this property, specialized hardware architectures and algorithms must be designed. Approximate computing investigates the inherent resistance of some applications to errors or uncertainties in their computations.

The approximate multiplier, a key element of approximate computing, is important for creating high-performance, energy-efficient digital systems. In numerous applications, such as signal processing, image and video processing, machine learning, and scientific simulations, the multipliers are used as fundamental arithmetic units. An approximate multiplier is different from a conventional multiplier in that it is not necessarily required to yield precise results.

The inherent error resistance of specific systems, by using approximate multipliers reduce the processing complexity, power usage, and latency. They achieve this by employing techniques like lower precision arithmetic, probabilistic logic, and simpler circuitry. Introducing controlled errors with tolerable limits, these approximations reduce computing precision and waste of resources.

There are various benefits of using approximate multipliers into computing technologies. Firstly, they enable significant power savings by using reduced-precision arithmetic logic. Less bits are used for calculation and representation, resulting in lower leakage power and dynamic power consumption. The second

benefit of approximate multipliers is that they can significantly reduce the area and delay of circuits, which enhances performance. The circuitry has been simplified, resulting in lesser transistors, interconnects.

Approximate multipliers have also seen to be effective in situations where minor computation errors can be tolerated without significantly affecting system performance. Examples include machine learning techniques, picture and video processing, and error-tolerant scientific simulations. By carefully applying approximate multipliers in such applications, it becomes possible to achieve significant increase in energy efficiency and performance while maintaining a decent level of accuracy.

The area of approximation computing is explored in this thesis, with a particular emphasis on the function of the modified approximate multiplier.

Researchers have investigated a number of techniques for producing accurate and efficient approximate multipliers in order to help the growth of approximation computing. Truncated arithmetic, stochastic computing, and algorithmic approximations are some of these methods. One method for efficiently computing approximations is stochastic computing, which depicts numbers as probabilities or bit streams. On the other hand, trumped arithmetic uses fewer bits than traditional arithmetic to trade off efficiency for accuracy. Algorithmic approximations include modifying algorithms to include controlled approximations at various stages of computation.

Despite the numerous benefits and advancements in approximate computing and approximate multipliers, challenges remain. One key challenge is finding the right balance between accuracy and performance. Despite the fact that approximate computing can considerably boost performance and energy economically, it is

crucial to establish the appropriate level of approximation for each individual application. The requirements of the application are acceptable error boundaries, and trade-offs in terms of power consumption, area, and performance must all be carefully analyzed and understood in order to achieve this balance.

The development of efficient design methods for approximation multipliers is a subject area that is still being researched. To improve the design process and produce the best results, this method is included in optimization algorithms, machine learning, and statistical modeling. By facilitating effective design space exploration, these techniques seek to solve the complexities of approximate multiplier design.

In conclusion, approximation computing and approximate multipliers present a viable route to high-performance and energy-efficient computing systems. Approximate multipliers enable power savings, reduced circuit complexity, and performance enhancement by introducing regulated approximations. They are particularly useful in applications where without harming the system's general functionality, minor errors can be tolerated. The goal is to investigate the field of approximation computing, with a particular emphasis on the significant function of approximate multipliers. By examining the design methods, trade-offs, and challenges of approximate multipliers, it is hoped to contribute to the growing body of knowledge in this area and stimulate further research into the subject of approximation computing.

1.1 THESIS ORGANIZATION

Several approximation architectures are used in order to minimize output errors while significantly improving area, latency, and power consumption. Stages are compressed using a variety of approximate adders and inexact compressors to achieve a fast processing approximation multiplier. FPGA implementations of these modified architectures are presented in terms of area, latency, and power.

There are four chapters in this thesis, including an introductory chapter. Chapters are organized as follows.

Chapter 1: Introductory part along with literature survey, aim and scope of the proposed work.

Chapter 2: Describes the several unsigned multipliers such as Booth Multipliers, Array Multipliers, Wallace Multipliers and Dadda Multipliers with different dimensions.

Chapter 3: Provides the outline of using exact compressor, approximate full adder and half adder along with the concept of using them to design an approximate multiplier of different dimensions.

Design and implementation of existing and modified unsigned approximate multipliers have been performed with the help of Xilinx Vivado 2022.2 synthesis tool in FPGA platform. Also, the comparison of the effectiveness of various approximate multipliers and existing approximate multiplier architectures in terms of area, delay, and power consumption are presented.

Chapter 4: Concludes the thesis with some potential areas for future research.

1.2 LITERATURE SURVEY

Digitization helping in pushing the boundaries of predefined circuits and trying to achieve more compact, fast processing units with more battery life to give longevity of the device application. Due to digitization digital platforms are facing issues with storage and transmission enhanced data qualities with huge memory storage. Several researchers have already been introduced different types of unsigned and approximate multipliers.

Reddy et al. [14] the design of a unique approximately 4:2 compressor is suggested along with Dadda Multiplier's redesigned architecture is presented. The efficiency of the suggested compressor and multiplier is assessed by experimental testing in a 45 nm standard CMOS technology, and their parameters are compared to those of the most advanced approximation multipliers. When compared to previous approximate compressors described in the literature, the suggested compressor significantly reduces error rates. Additionally, the proposed multiplier reduces power consumption, latency, and area by 35%, 36%, and 17%, respectively, as compared to those of the precise multiplier. Some image processing applications evaluate the multiplier's effectiveness. The suggested multiplier operates on images that are 85% structurally similar to the exact image output.

Ansari et al. [15] introduced a 4:2 compressor which exhibits a significant number of erroneous outputs. After its signals and inputs were encoded, these mistakes were decreased. These compressors created greater scale multipliers, which are used in the interface nulling computations of MIMO baseband receivers, where cancelled codes cancel out channel noise. Error distance variability and root mean square error distance are less precise multipliers, compared to a 16 bit LOD Mitchell approximation multiplier. Image sharpening is performed using a logarithmic MAC.

Hashemi et al. [2] proposing various innovative techniques in areas of hardware approximate computing. Those are mostly implemented on deep neural networking and iris recognition system using stimulations and synthesis of industry standard call libraries. Improvement in error bound helps in balancing amount of positive and negative errors. Proposed both dividers and multipliers with energy trade off within five percent error range.

Anushaa et al. [12] designed an approximate adder and synthesized the circuit by applying it with 8 bit dada multipliers. Circuit's quantitative analysis is presented presented in terms of error distance, error rate, mean error distance and normalized error distance. Circuits are used for image blending and sharpening purposes. These are used for image quality calculation.

Adams et al. [7] the drawbacks of transistor shrinking as well as the advent of approximation computing as a fresh paradigm for performance enhancement. The emphasis is on hardware for approximation arithmetic, and two approximate restoring dividers and three approximate Booth multipliers are presented and assessed for correctness and hardware performance. When compared to exact designs, the multipliers and dividers produce significant power savings, with the ABM-M3 and AXRD-M2 seeing the largest reductions in power usage. These rough hardware designs are used in a variety of applications, including image transformation, matrix multiplication, and a FIR filter implementation, where they outperform benchmarking models in terms of performance. Based on the ABM-M3 concept, a new approximate multiplication architecture called ABM-MAC was created. ABM-MAC achieves a 42% area reduction and a 33% power reduction in comparison to the precise design, outperforming other inexact MAC units in terms of accuracy and hardware performance. ABM-MAC's main benefit is its excellent accuracy, despite the fact that its hardware performance is competitive when compared to

benchmarking models. ABM-MAC's efficacy is confirmed by convolution operations employing 55 and 77 kernels.

Edavoor et al. [1] new 4:2 compressor designs are suggested. Using image processing techniques like image multiplication and smoothing, researchers confirmed the suggested designs.

The first design is a high-speed and area-efficient compressor that, when compared to other compressor designs now on the market, significantly reduces area, delay, and power consumption. The proposed design maintains comparable accuracy with a 25% error rate and a 1 absolute error deviation in both the positive and negative directions. This decrease in area, latency, and power consumption allows for a reduction in the MED and MRED without altering the error rate.

A modified dual-stage compressor is the second design that has been suggested, and it further optimizes the metrics for area, latency, and power usage without reducing accuracy. Using 45-nm technology and a 1V supply voltage, the suggested designs were implemented at the transistor level.

Overall, the paper presents 8bit and 16bitDadda multiplier circuits using mentioned innovative designs for 4:2 compressors that can improve the performance and efficiency of digital signal processing and arithmetic operations in various applications.

Moaiyeri et al. [16] made an approach for designing 4:2 compressor using FinFET and Quantumdot Cellular Automata(QCA). Applications for image processing use the proposed design. The quality metrics of image is processed by MATLAB. Peak signals to noise ratio is calculate for image quality evaluation. This compressor uses low-power nano circuitry that is accurate enough.

Ramasay et al. [5] Proposed Approximate adders do not produce accurate outputs. A modified approximate full adder at the gate level of logic is used for the relaxation of numerical exactness benefit. In order to reduce an area's complexity, the conventional adder's sum phrase is updated to use the carry-based approximation adder (CBAA) in place of the essential XOR operation. To demonstrate this concept, a variety of complete adders of the approximate approximation type with decreased complexity at the gate level were developed. When compared to current implementations using accurate adders, simulation results show that adopting the suggested approximate adders can save up to 98% of the power usage.

Maddisetti et al. [17] Compressors are now necessary parts of multiplier designs for partial product reduction stage. Previously, various adders, such as carry save adders, were used to partially reduce the product, but due to the need for low power and a smaller footprint, adders were replaced by order compressors, such as 3:2, 4:2, and 5:2.

Proposed work provides an efficient and accurate 4:2 compressor with inexact logic minimization by flipping some of the output bits. An 8 bit Dadda multiplier has been used to implement the proposed 4:2 compressor, and the average power, area, and propagation delay of the architectures have been calculated. The spectre simulator was used for all simulations in the 45nm technology node. Error analysis has been carried out using MATLAB to determine the difference between the exact and approximate proposed circuits. This paper's application idea is to upload and download the roughly 4:2 compressor that Cadence Virtuoso has implemented.

Pathak et al. [18] using a simulation of a nearly full adder, the work assesses the analysis of different multipliers. Due to the fact that it allows for a significant decrease in space and power consumption with some accuracy loss, the approximation approach is frequently utilized in digital signal processing and

multimedia applications. Since little flaws are invisible to the human eye, they have no effect on the visual quality. This approach offers the benefit of minimizing device usage for such applications, which ultimately leads to a decrease in both transmission speed and power consumption. The redundant bit will be lowered, along with the amount of storage needed to keep the data. It is possible to reduce its computational complexity by the approximation of arithmetic circuits without suffering a substantial impact on coding.

Saraswati et al. [13] The power consumption, latency, and area have all been greatly reduced by approximate computing of the DADDA multiplier. As a result of the significant space, power and area requirements of a digital signal processing application, where multiplication is a key function, we choose an approximative multiplier to simplify the system. In addition to developing and evaluating about 4:2 compressors, we also analyze and design an 8bit Dadda. When multiplying exactly, we use 18 compressors, and when multiplying approximatively, we use 17 compressors.

Chandaka et al. [19] The modified Wallace Tree Multiplier (MWTM), cut down on the calculation time and power delay product, approximate multipliers were developed. However, it offers a strong prospect for significantly enhancing the area, power and for partial product addition in multipliers using two modified approximate 4:2 compressors. It has been found that the Normalized Error Distance (NMED), Mean Relative Error Distance (MRED), and Power Delay Product (PDP) are decreased while using the suggested MWTM. Utilizing 90-nm CMOS standard cells, the suggested architectures are created. Wallace tree multipliers that have been modified are created in sizes ranging from 8 bits to 32 bits, and the effectiveness of these multipliers is compared to the performance of current general multipliers. In comparison to conventional multipliers, the 8-bit MWTM's synthesis results provide an average reduction in latency and power of between 10% and 55.37% and 13.03% to 13.78%.

Additionally, 16-bit MWTM demonstrates that the average reductions in latency and power are between 0.11% and 3.12% and 0.28% and 6.59%, respectively. Additionally, 32-bit MWTM demonstrates that the average power reduction ranges from 8% to 27.99%. The suggested MWTM is used to implement the image processing activities of image blending, picture smoothening, and edge detection. The outcomes demonstrated the MWTM's effectiveness.

Ranjbar et al. [20] using three new 4:2 approximate compressors, a novel 8-bit approximate multiplier is proposed. Its delay and error are lower than those of multipliers built using conventional 4:2 approximate compressors, and it also has a lower delay than an 8-bit multiplier built using 3:2 precise compressors. Each innovative compressor in the multiplier is built with an output carry that is distinct from that of its predecessor. As a result, the carry propagation delay issue is solved, and a quick multiplier is built. The evolutionary algorithm is used to select the optimal compressor from the three suggested compressors for each multiplier's column in order to produce the multiplier that is the most accurate. Additionally, for additional error reduction, one can only utilize the approximate compressors at the k least significant multiplier's columns. For image blending and compression, the recommended multiplier is employed. Our simulations demonstrate that, for instance, the error and latency of the suggested technique for k=9 are at least 32.52% and 33.10% fewer than those of conventional 4:2 approximation compressor based multipliers, respectively.

Sudha et al. [21] the key component of high-performance computing systems is the construction of arithmetic circuits. Particularly the arithmetic operation of multiplication is one that is frequently employed. Signal processing and other scientific applications, multiplication is a widely utilized arithmetic operation.

Interest in better speed, lower cost and less area are the key reasons for multiplication in a hardware-intensive operation. Comparing the multipliers, it can be seen that the Dadda multiplier contains three phases for partial product

multiplication and a set method for reducing the parameters. Using two distinct novel approximate 4-2 compressor designs, the approximate compressor for the dadda multiplier was created. Real-time applications use image multiplication. The use of image multiplication is executed utilizing this multiplier and Xilinx.

Rao. B et al. [23] for digital processing, approximate computing is a compelling paradigm. Computer arithmetic designs find imperfect computing to be particularly intriguing. The two new, about 4:2 compressors that are used in a multiplier are the subject of this project's novel design. These designs rely on various compression characteristics, allowing calculation errors to compensate for a design's circuit-based figures of merit.

It is suggested and examined for a Dadda multiplier two possible techniques for applying the proposed approximation compressors. In-depth simulation results are shown, and a Field Programmable Gate Array (FPGA) is used to create the Dadda multiplier utilizing approximation compression in hardware.

Ramasamy et al. [22] VLSI multiplier circuits approximation computation errors are growing as technology scales. It is crucial because approximation of operands is the most popular technique for executing multiplication results quickly and efficiently. However, image processing programs cannot use this conventional approximation. There are two novel designs for image compression are proposed: enhanced HSAM and high accurate hybrid segment approximate multiplier (HSAM). Some exact applications are, the current static segment method-based approximate multiplier is not suitable, while for cost-effective applications, the existing dynamic segment method-based approximate multiplier is not acceptable. Enhance accuracy and cost efficiency, the suggested work combines the benefits of static segment method and dynamic segment method. The suggested approximation multipliers HSAM88 and EHSAM88 offer 99.85% and 99.999% accuracy for various inputs, respectively, with only a slight increase in overhead area. The suggested HSAM

uses less energy, and without any area overhead, the suggested EHSAM uses less energy. In comparison to the current SSM88 technique, the suggested HSAM and EHSAM have speed improvements of 40% and 85%, respectively.

Krishna et al. [26] for image and video processing applications, an energy-efficient approximation multiplier is created using an approximate compressor. In comparison to the existing designs from the literature, this research suggests a roughly 4:2 compressor design with an 18.75% error rate that uses 15% less energy on average. Two multiplier variations are suggested in this study, one with solely approximate compressors and the other with both approximate and exact compressors.

Reddy et al. [24] in this research, we offer a novel truncation strategy and an error correcting term that are applied to a recursive multiplier architecture. Additionally, the area, latency, and power are greatly reduced thanks to a truncation method and a correction term that account for the approximate multiplier's mistake. The suggested multiplier is then tested against an image sharpening method. The results of the simulations clearly show that the proposed multiplier outperforms the currently used multipliers.

Rani et al. [25] In order to maximize the area, delay, and power without compromising the accuracy metrics, this research suggests new structures for approximate multipliers. The speed of the adders and the amount of energy used by the processor have a significant impact on its overall performance. In order to increase the effectiveness of the approximate multipliers, two different types of compact error-tolerant approximate adders are created and employed in this paper. In comparison to the current architectures, the proposed approximation multipliers perform well in terms of area, latency, power, and accuracy. The performance of the error-tolerant adder is assessed in the real world with the aid of the image blending application. The modeled designs are evaluated using peak signal-to-noise ratio (PSNR) performance and the structural similarity

index metric (SSIM). The estimated multipliers and adders that are suggested exhibit performance better in terms of PSNR and SSIM, and it has been discovered that they are an optimal design that may be used successfully in a variety of error-tolerant image processing applications.

Gowdar et al. [28] The design of two 88 approximate multipliers based on new approximate 3:2 and 2:2 compressors is presented in this research article. The proposed "approximate Wallace multiplier" (AWM) is a term for the multipliers that are derived based on the Wallace multiplier architecture. According to "Design Metrics" (DMs) such power, delay, "power-delay-product," and area, the effectiveness of these proposed AWMs has been evaluated and examined. All the multipliers under consideration have been specified using Verilog code and generated using Cadence's "RTL Compiler" (RC) tool utilizing a 180 nm standard cell library in order to extract these DMs. The outcomes of the synthesis reveal that the suggested AWMs achieve a superb performance in terms of DMs. Additionally, in an image processing application, the peak signal-to-noise ratio (PSNR) of the AWMs and other Wallace multipliers that were constructed based on reported approximation compressors has been examined. The comparison findings reveal that the proposed multipliers have a better PSNR (more than 50 dB).

Zhao et al. [27] This study proposes a novel approximate multiplier that, by utilizing OR and AND gates, can reduce the complexity of multiplication while improving area and power performance. Design parameters are compared with exact multipliers and recently proposed approximate designs to assess the effectiveness of the proposed approximate multiplier. The power usage of the proposed approximation multiplier is only 32% that of the precise multiplier, according to the findings of the experiments. Further testing of the proposed approximate multipliers is done in edge detection and image sharpening applications. Our suggested approximate multipliers' peak signal to noise ratio

(PSNR) and structural similarity (SSIM) demonstrate their use in image processing.

Pei et al. [42] this article proposes three innovative approximations 4:2 compressors that can be used in 8-bit multipliers. The error performance of the approximation multiplier improves with the suggested 4:2 compressors, an error-correcting module (ECM) is presented in the meantime. The estimated 4:2 compressor's outputs are ingeniously reduced to one, which enhances energy economy even more. The simulation results show that the suggested approximation compressors UCAC1, UCAC2, and UCAC3 accomplish 24.76%, 51.43%, and 66.67% reductions in delay, 71.76%, 83.06%, and 93.28% reductions in power, and 54.02%, 79.32%, and 93.10% reductions in area, respectively, as compared to the exact 4:2 compressors. Additionally, the average power consumption is reduced by 49.29% when these suggested compressors are used in 8-bit multipliers.

Yi et al. [53] for error-resistant multiplication with a low supply voltage, a novel approximation 4-2 compressor and its circuit implementation are proposed in this study. When compared to various multipliers for the operand length of 8 bits, simulation results show that the approximate multiplier with our suggested approximate 4-2 compressor consumes the least energy per operation while maintaining the same computational accuracy. When compared to precise multiplication, it delivers a 26.7% reduction in energy-delay product (EDP).

Ha et al. [44] an approximate 4:2 compressor can be used to produce power-efficient circuits for approximate multiplication. A unique design is presented that modifies an existing approximate 4:2 compressor design and includes an error recovery mechanism. Comparing to other proposed 4:2 compressor based approximate multiplier systems, the proposed design is more accurate, requires less hardware, and uses less power even with the added error recovery module.

H. Afzali-Kusha et al. [54] This article examines a Dadda(X) multiplier with adjustable accuracy and energy efficiency. The structure uses approximate width setting and voltage overscaling as approximation knobs to increase multiplier reliability and lifespan while reducing energy usage. The latter may only be activated in the design time, but the former may be set both in the design time and the runtime. A specific accuracy level, for the over scaled voltage and partial product columns for energy optimization are identified. Voltage over scaled columns typically feature larger switching activity and lower bit significances in order to keep the error within a reasonable range.

A minimal overhead realization, the structure uses a small number of level shifters. The approximate columns that follow the first column are all connected. The output of the multiplier may additionally be truncated by four bits to increase multiplier efficiency even more. A 15-nm FinFET technology is used to examine the effectiveness of the Dadda(X) structure. The results show that when the approximation mode with the mean relative error distance (MRED) of 0.11 is taken into account, up to 43% energy savings are made. Furthermore, in this instance, the multiplier's delay degradation caused by bias temperature instability (BTI) is reduced up to 9.9% as opposed to 50% in the precise mode. Additionally, it is investigated how process variables affect the Dadda's(X) accuracy. Lastly, it is determined how well the Dadda(X) multiplier performs when applied to neural networks for image classification and image-processing tasks.

Ahmadinejad et al. [45] this paper’s, new approximation compressors with improved energy and quality efficiency are proposed. The complemented partial products generated using NAND gates, which uses less transistors. New approximation compressors are also being developed with varying levels of precision and performance. Hence, three hybrid approximate multipliers are developed that offer various trade-offs between accuracy and hardware

efficiency. The 7nm FinFET model is used as a cutting-edge technology to simulate the proposed designs using HSPICE. Additionally, MATLAB is used to assess the effectiveness of the approximative multipliers in neural network and image processing applications. The new designs can be thought of as effective substitutes for the precise multipliers in neural network and image processing applications since they offer significantly better quality and energy metrics compromises than the earlier solutions.

Strollo et al. [55] the scientific literature suggests a number of circuits constructed with about 4:2 compressors, although approximate multipliers draw a lot of attention. In this study, a thorough analysis and comparison of the approximate 4:2 compressors with that have been previously proposed in the literature. A total of twelve unique approximate 4:2 compressors, has been propose a novel approximate compressor. CMOS technology in 28nm, 8bit and 16bit multipliers are designed using the researched circuits. It is examined two multiplier setups, both signed and unsigned, with various degrees of approximation for each operand size. Our analysis shows that the optimum approximation compressor architecture depends on the needed precision, the multiplier is signed, and the taken into account error metric, and that there is no single best solution.

Towhidy et al. [58] for applications that are error-tolerant, approximate multipliers offer power and space savings. The first two approximate floating-point multipliers suggested in the paper are floating-point pseudo-Booth (PB) and floating-point iterative pseudo-Booth (IPB), both of which are based on two-dimensional pseudo-Booth encoding. Iteration, encoder's radix and word length after truncation are three factors that can be adjusted to adjust the accuracy of suggested multipliers. Then, in order to reduce multipliers' power consumption, they created standard iterative multipliers with a streamlined steering circuit for their correction section. A streamlined steering circuit for the

floating-point area is used to implement the suggested iterative multipliers, which are contrasted with traditional iterative integer multipliers. The findings show that in comparison to the precise floating-point multiplier, the suggested PB-R4-W4 and IPB-R16-W19 offer up to 98.9% and 67.5% savings in power usage for TSMC 180nm CMOS technology, respectively. They also had respective MRED values of 2.9% and (7.4×10−4)%. Finally, we tested the effectiveness of the suggested multipliers in two image processing tasks smoothing and sharpening as well as a hyper-plane classifier.

Lee et al. [35] the revolutionary approximate adder described in this literature uses a lower halve approximation that is simplified to save energy and power. While delivering a performance with adequate precision, the suggested technique lowers hardware costs. The suggested adder reduces area and power by 67% and 91%, respectively, when implemented in 32-nm CMOS technology as compared to a traditional adder. The power-delay and energy-delay products are improved by 13.1% and 17.0%, respectively, in terms of energy when compared to the other approximation adders taken into consideration. Furthermore, the proposed adder exhibits outstanding energy economy while producing output quality that is extremely promising when used in a digital image processing application as compared to an exact adder.

Vahdat et al. [57] a scalable approximate multiplier known as the truncation- and rounding-based scalable approximate multiplier (TOSAM) is introduced. It decreases the number of partial products by truncating each of the input operands according to their leading one-bit position. The proposed architecture uses modest fixed width multiplication, add and shift operations to execute multiplication instead of the exact multiplier which results in significant reductions in energy usage and area occupied. Rounding the input operands of the multiplication step to the nearest odd number will increase overall accuracy. As a result of input operands being trimmed depending on their leading one-bit

places, accuracy is no longer strongly correlated with input operand width, and the multiplier can be scaled. The input operand widths expand, greater gains in design parameters (such as area and energy usage) can be made. The design parameters of the proposed approximate multiplier are compared to those of an exact multiplier and a few other recently suggested approximate multipliers in order to assess its effectiveness. Results show that compared to the precise multiplier, the suggested approximate multiplier decreases latency, area, and energy usage by up to 41%, 90%, and 98%, respectively. The proposed approximate multiplier has a mean absolute relative error in the range of 11%-0.3%. Additionally, it performs better in terms of speed, area, and energy usage than other approximation multipliers. The proposed approximate multiplier has an error distribution that is practically Gaussian and a mean value that is very close to zero. Use of it in the design of a JPEG encoder as well as applications for classification and sharpening. The outcomes show that there is little any output quality degradation. Additionally, an accuracy-configurable TOSAM proposed, where the energy consumption of the multiplication operation can be changed in accordance with the minimal precision necessary.

Seo et al. [40] an innovative hybrid error reduction technique combined with an energy efficient approximate adder to greatly increase computation accuracy while incurring very little additional power and space overhead. Two input bits are required for the proposed hybrid error reduction system and the approximation outputs are modified to reduce the error distance, improving accuracy all around. When used in 65-nm CMOS technology, the suggested design has in comparison to traditional accurate adders, energy, power and area efficiencies that are 3, 2 and 1 respectively, three times higher. In terms of accuracy, the proposed hybrid error reduction strategy enables the proposed adder's error rate to drop to 50%, compared to the lower-part OR adder's error rate of 68% and the optimized lower part OR constant adder's error rate of 85%,

respectively. Additionally, compared to the other approximation adder taken into consideration in this research, the suggested adder performs up to 2.24, 2.24, and 1.16 times better in terms of the mean error distance, normalized mean error distance (NMED) and mean relative error distance respectively. Importantly, the suggested adder is discovered to be the most competitive approximate adder when concurrently assessed in terms of the hardware cost and computation accuracy. This is due to an effective design tradeoff among latency, power, energy and accuracy. In comparison to existing approximation adders, our suggested adder reduces the power, energy and error-delay product NMED products by 51%, 49% and 47% respectively.

Ansari et al. [56] a neural networks (NN) accuracy typically necessitates the use of more energy-intensive, bigger hardware. However, to cut down on implementation costs, approximation computing methods can be used due to the error tolerance of NNs and the applications they are used in. More cost-effective approximation multipliers (AMs) can drastically lower hardware costs because multiplication is the greatest resource- and power-demanding function in NNs. This article, demonstrates how adding noise through the use of AMs can enhance NN accuracy.

Focusing on two types of AMs: AMs with a conscious design and AMs built using Cartesian genetic programming (CGP). In order to assess their impact on the classification accuracy of the Mixed National Institute of Standards and Technology (MNIST) and Street View House Numbers (SVHN) data sets, respectively, the exact multipliers in two representative NNs, a multilayer perceptron (MLP) and a convolutional NN (CNN), are replaced with approximate designs. Intriguingly, reductions of 71.45% and 61.55% in the energy consumption and area, respectively, lead to improvements in classification accuracy of up to 0.63%.

Finally, the AM traits that tend to favor one design over another in terms of NN accuracy are identified. Then, a predictor that shows how effectively an AM is expected to function in a NN is trained using those attributes.

1.3 AIM

The goal of approximate multipliers has become a strong research area in the field of VLSI (Very Large Scale Integration) technology. Designers are experimenting with creative methods to establish a compromise between precision and computing overhead as semiconductor technology continues to scale down and the demand for power-efficient circuit increases. In contrast to conventional designs, approximate multipliers offer significant benefits in terms of area, power, and speed. They also allow for regulated output errors. The main goal is to take use of an application's inherent error resilience. For example, picture and signal processing systems can tolerate small errors without significantly degrading output quality. These multipliers take advantage of approximation to provide increased performance, reduced silicon footprint and greater energy efficiency, making them a viable option for a variety of future VLSI systems. To make sure that the introduced imperfections stay within acceptable bounds, however, the search of efficient approximation multipliers necessitates careful consideration of application-specific needs and error analysis.

1.4 SCOPE

Very Large Scale Integration (VLSI) technology integration has become increasingly important in recent years for the advancement of contemporary computing systems. As varied applications are required, there is a constant search for efficient and high-performance circuits. Multipliers as one of the many building blocks found in VLSI circuits, are of utmost significance in computationally demanding activities including digital signal processing, image processing, and machine learning algorithms. A viable solution to the growing demand for high-throughput, energy-efficient multiplication operations is the development of approximation multipliers.

The use of approximate multipliers in VLSI technology is dependent on the accuracy and performance trade-off. A controlled insertion of errors during the multiplication process is possible using approximate multipliers, as opposed to traditional exact multipliers. This method takes advantage of the inherent resilience of many applications to accept small errors, yielding considerable speed and power consumption reduction advantages. Additionally, the configurability of approximate multipliers enables designers to adjust the degree of approximation to meet the needs of a given application.

Additionally, approximate multipliers show promise for use in specialized fields like approximate computing and error-tolerant software. Researchers can open up new possibilities for improving efficiency and utilizing error resilience for particular algorithmic tasks by selectively using approximation multiplication in these areas. The use of approximate multipliers is anticipated to increase as VLSI technology develops, opening up new possibilities for innovation in design processes and circuit optimization techniques.

To make sure that errors introduced by approximate multipliers do not impair overall system performance or result in negative consequences in safety-critical applications, it is crucial to strike a fine balance between accuracy and approximation level. In order to fully realize the potential of approximate multipliers in VLSI technology, extensive research is therefore required to investigate novel design strategies, error mitigation techniques, and effective hardware implementations, making it a compelling avenue for ongoing research and development in the field of digital circuit design.

CHAPTER 2

MULTIPLIERS

MULTIPLIER

A variety of approaches are proposed to achieve less LUT with high speed performance. Different types of compressors are designed to reduce processing stages of a multiplier for fast processor. Compressors like dual stage compressors and many effective parallel operations are proposed to part partial products in column compression to accelerate the speed. In compressors mostly approximate adders instead of exact adder to improve on delay, area consumption and energy consumption. Some research is done on transistor level and CMOS process technology. Approximate compressors are building blocks of approximate multiplier.

2.1 BOOTH MULTIPLIER:

Booth Multiplier is a hardware-based algorithm used for fast and efficient multiplication of binary numbers. The Booth's multiplication multiplies two signed numbers in the complement of two. Relative to more straightforward algorithms, this method utilizes fewer additions and subtractions.

There are numerous ways to implement Booth's algorithm. In order to design this experiment, a controller and a data path were used. The control signal that the controller sends controls the actions that are performed on the data in the data path. The data stream contains registers for multiplier, multiplicand, intermediate results of ALU, adder/ subtractor, counters, and other combinational units. The Booth's multiplier which multiplies two 4-bit values. The data processing unit in this instance is an adder and subtractor device. Let M, Q and A are 4-bit registers, where A holds the outcomes of the adder unit, Q is the multiplier, while M is the

multiplicand. A down counter is the one that keeps track of how many operations are necessary for multiplication. The five control signals produced by the controller regulate the data path's data flow. These signals include load (to load data into registers), add (to start an addition or subtraction operation), sub (to start a subtraction or right shift operation), shift (to start an addition or right shift operation), and dc (to start a counter decrement). The control signals are produced by the controller in accordance with the data path’s input. Low bits can then be shifted out, allowing subsequent additions and subtractions to be performed only on the highest bits. Strings of the multiplier are typically described as being converted by the algorithm to high-order +1 and low-order 1 at the endpoints of the string. Because there is no high-order +1 when a string traverses the MSB, the result is that the correct value is interpreted as being negative. Booth encoding schemes, include the Booth's Radix-2, Radix-4, and Radix-8 methods.

A comprehensive comparative study is conducted, pitting the Booth multiplier against conventional multiplication algorithms, such as the array multiplier and Wallace tree multiplier. The analysis encompasses performance metrics, area requirements, and power consumption to identify the strengths and weaknesses of each approach.

2.2 ARRAY MULTIPLIER:

Two binary values are multiplied using an array of full and half adders in a digital combinational circuit known as an array multiplier. The 2's complement number system is used to express multiplicand and multiplier because they can both be positive or negative. Similar methods can be used if the multiplier operand is positive, but care must be made with sign bit extension. All numbers that are added

in one adder stage when carrying save arithmetic is used must all have the same bit length. Using this array, the several product keywords involved are added virtually simultaneously. An array of AND gates is used to construct the different product terms before the adder array. In order to save time and increase area, carry save adders are employed in place of CRAs. These adders pass each carry and sum signal to the adders of the stage behind them. Any rapid adder (often a carry ripple adder) can be used in a final adder to produce the final product. A sequential process involving bit-by-bit inspection of the multiplier's bits and generation of partial products requires a sequence of add and shift micro-operations. The addition is performed both serially and simultaneously. When multiplying an array, we must add as many partial products as there are multiplier bits. Array multiplier's have delay in computation and very high power usage. Larger areas are produced by additional digital gates. If AND gates are used for the product, summation is carried out using full adders and half adders, where the partial product is shifted in accordance with their bit ordering. n*n AND gates compute the partial products in a n*n array multiplier, and n* (n - 2) full adders and n half adders can be used to add the partial products. The MSB bit of the partial product is the bit on the right. When multiplying, the partial products are now relocated to the left and added to produce the whole result. Repeat this procedure until no more partial products are left enough for adding. There are 8 inputs and 8 outputs on the 4bit array multiplier displayed. Two binary values can be multiplied in a single micro-operation by a combinational circuit that generates the product bits all at once.

Since all it takes to multiply two numbers in this way is for the signals to travel through the gates that make up the multiplication array, it is a quick method. An array multiplier, however, needs a lot of gates, therefore before the invention of integrated circuits, it was not cost-effective. Consider multiplying two 2-bit values

for array multiplier implementation with a combinational circuit. Applications of array multipliers are filtering in the Fourier transform, and picture coding are arithmetic operations that are carried out using an array multiplier and high-speed performance. Because this multiplier lacks a sign bit expansion, it deals with signed numbers erroneously. There is a technique to fix this error without having to enlarge every bit in the partial product addition. The multiplier circuit's base is the add-shift algorithm. The main advantage of the array multiplier is that it has a simple and predictable structure. A substantial latency and high power consumption are disadvantages of an array multiplier.

2.3 WALLACE MULTIPLIER:

Wallace multiplier is a widely used technique in digital circuit design that significantly improves the performance and efficiency of multiplication operations. Practical applications of Wallace multipliers in approximate computing, which aims to trade off accuracy for improved efficiency in various computational tasks.

The Wallace multiplier is based on a tree-based reduction algorithm that optimizes the partial product generation and accumulation process. By employing a modified booth encoding scheme, the Wallace multiplier reduces the number of partial product terms and achieves higher compression ratios compared to conventional multipliers. This reduction in terms results in reduced hardware complexity, lower power consumption, and improved overall performance.

In the realm of approximate computing small deviations from exact results are tolerable, the Wallace multiplier finds significant utility. Applications such as image and video processing, where slight variations in pixel values are often imperceptible to human observers, the Wallace multiplier can be employed to

perform approximate multiplication operations. Leveraging the efficiency gains of the Wallace multiplier, these computations can be executed more rapidly, making it suitable for real-time applications.

Furthermore, the Wallace multiplier can be integrated with other approximate computing techniques, such as algorithmic approximations and error-tolerant architectures, to achieve a comprehensive system-level approach. This combination enables a fine balance between computational accuracy and energy efficiency, making it particularly relevant in resource-constrained environments like mobile devices and Internet of Things (IoT) applications.

2.4 DADDA MULTIPLIER:

Digital arithmetic multipliers play a vital role in various computational tasks, ranging from signal processing to scientific simulations. The Dadda multiplier, a well-established technique, has gained considerable attention due to its high-speed and power-efficient characteristics. Dadda multiplier is an algorithmic method for multiplying binary numbers that significantly reduces the number of partial products required for the final result. It achieves this by exploiting the inherent structure of multiplication and optimizing the number of adders needed to perform the partial product additions. Concept behind the Dadda multiplier is to employ a tree-like structure, referred to as a carry save adder (CSA) tree, which reduces the complexity of the circuit.

One of the significant advantages of the Dadda multiplier is its ability to deliver high-speed multiplication operations. Reducing the number of partial products and optimizing the carry propagation, the Dadda multiplier minimizes critical path delays and enhances the overall performance of the circuit. Consequently, it has

become a demanding choice for fast arithmetic operations, such as digital signal processing and image processing.

Furthermore, the Dadda multiplier offers notable power efficiency benefits, by reducing the number of adders and minimizing the switching activity within the circuit. It decreases the power consumption associated with multiplication operations. This characteristic is particularly desirable in energy-constrained systems, such as mobile devices and Internet of Things (IoT) applications, where power efficiency is a critical concern.

In the context of approximate computing, the Dadda multiplier has proven to be a valuable tool. Approximate computing involves trading off accuracy for performance or power consumption in applications where a slightly imprecise result is acceptable. Leveraging the Dadda multiplier's inherent optimization techniques, approximate computing frameworks can achieve substantial gains in speed and power efficiency.

For example, in image and video processing applications, where minor errors in pixel calculations may not be visually perceptible, approximate computing can be employed to speed up the overall processing time or reduce power consumption. The Dadda multiplier, with its high-speed and power-efficient characteristics, can serve as a fundamental building block within approximate computing frameworks, enabling significant improvements in performance and energy efficiency.

Dadda multiplier offers a compelling solution for high-speed and power-efficient multiplication operations. Its inherent optimization techniques make it an attractive choice for various applications, ranging from digital signal processing to image processing. Furthermore, in the emerging field of approximate computing, the Dadda multiplier plays a pivotal role in achieving substantial gains in performance

and energy efficiency. Its utilization within approximate computing frameworks opens up new possibilities for accelerating computation while minimizing power consumption.

Let’s illustrate the practical use of the Dadda multiplier in approximate computing, let's consider an example from the field of machine learning. Many machine learning algorithms involve matrix multiplication, which is computationally intensive and can benefit from approximate computing techniques. In certain machine learning tasks, such as neural network training, achieving an exact result for every matrix multiplication operation may not be necessary. This acceleration can lead to significant time savings during training, enabling machine learning practitioners to experiment with larger datasets or deeper network architectures. Approximate computing techniques, including the utilization of the Dadda multiplier, can be employed to speed up the training process without significantly impacting the accuracy of the final model.

Moreover, the power efficiency of the Dadda multiplier is particularly advantageous in energy-constrained scenarios, such as edge computing devices or mobile platforms. By reducing the power consumption associated with matrix multiplication operations, approximate computing using the Dadda multiplier can extend the battery life of mobile devices and improve the energy efficiency of edge computing deployments.

In summary, the Dadda multiplier is a high-speed and power-efficient multiplier algorithm that has found practical applications in various domains. Its optimization techniques make it particularly suitable for approximate computing, where performance and power efficiency trade-offs can be leveraged. Incorporating the Dadda multiplier within approximate computing frameworks, significant gains in

speed and power efficiency can be achieved, benefiting applications ranging from digital signal processing to machine learning.

In addition to its practical applications in approximate computing, the Dadda multiplier also offers advantages in terms of scalability and hardware complexity. Its algorithmic approach allows for efficient implementation in hardware architectures, making it suitable for integration into large-scale computing systems.

Furthermore, the Dadda multiplier can be extended to support different radix sizes, enabling flexibility in the design of arithmetic units. This adaptability makes it a versatile multiplier that can be tailored to meet the specific requirements of different applications and computing platforms, by combining the high-speed and power-efficient characteristics of the Dadda multiplier with the flexibility of approximate computing, researchers and engineers can unlock new possibilities in performance optimization and energy efficiency across a wide range of applications.

In conclusion, the Dadda multiplier is a well-established algorithmic technique that offers high-speed and power-efficient multiplication operations. Its practical applications extend to the emerging field of approximate computing, where it can significantly enhance performance and power efficiency. By integrating the Dadda multiplier into approximate computing frameworks, researchers can explore new frontiers in speed optimization and energy-efficient computing, contributing to advancements in various domains, including digital signal processing, image and video processing, machine learning, and beyond.

CHAPTER 3

MODIFIED APPROXIMATE MULTIPLIERS

3.1 EXACT COMPRESSOR

Fast processing multiplier architecture includes high speed parallel operations with error resilience system. Approximation circuits must have minimal hardware complexity with maximum achievable accuracy. Process of calculating partial products in every stage is a time consuming and extending delay phenomenon, to overcome this compressor are introduced. A 4:2 compressors have five inputs and three outputs, of same binary weight. Main purpose of carry save addition is to reduce 'n' numbers of two numbers in the process. This design has a long critical delay of 4Δ, where Δ is the unitary delay through any gate in the design.

$$Sum = (A_1 \wedge A_2 \wedge A_3 \wedge A_4 \wedge C_{in})$$

$$Carry = C_{in} (A_1 \wedge A_2 \wedge A_3 \wedge A_4) + A_4 (\sim (A_1 \wedge A_2 \wedge A_3 \wedge A_4))$$

$$C_{out} = A_3 (A_1 \wedge A_2) + A_1 (\sim (A_1 \wedge A_2))$$

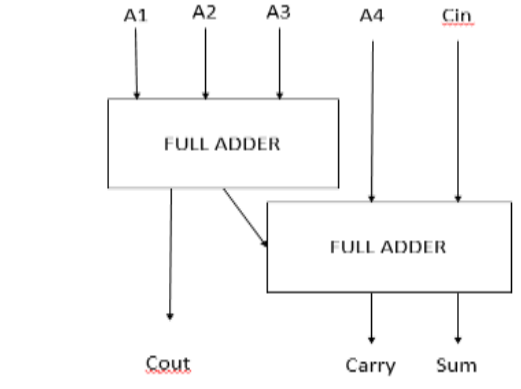

Fig 1: Exact Compressor

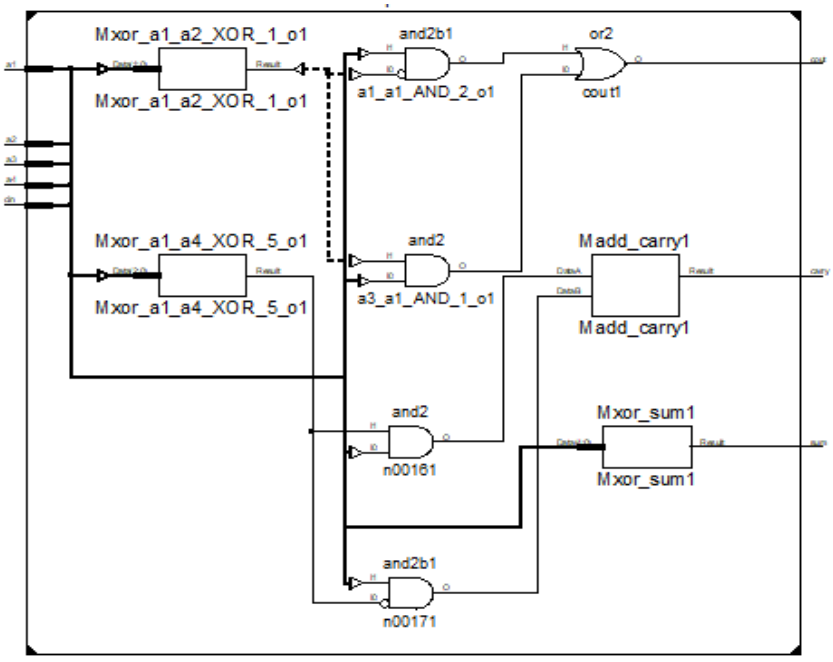

Fig 2: Gate level 4:2compressor

Table 1: Truth Table of 4:2 Exact Compressor

A1	A2	A3	A4	C_{in}	C_{out}	Carry	Sum
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	1
0	0	0	1	1	0	1	0
0	0	1	0	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	0	0	1	0
0	0	1	1	1	0	1	1
0	1	0	0	0	0	0	1
0	1	0	0	1	0	1	0
0	1	0	1	0	0	1	0
0	1	0	1	1	0	1	1
0	1	1	0	0	1	0	0
0	1	1	0	1	1	0	1
0	1	1	1	0	1	0	1
0	1	1	1	1	1	1	0
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	1	0
1	0	0	1	1	0	1	1
1	0	1	0	0	1	0	0
1	0	1	0	1	1	0	1
1	0	1	1	0	1	0	1
1	0	1	1	1	1	1	0
1	1	0	0	0	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	0	1	0	1
1	1	1	0	1	1	1	0
1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1

The 16-bit compressor is composed of two parallel 8:4 compressor. The 32bit compressor is composed of four 8:4compressor in parallel. The 16-bit compressor is marked by black outlined box in figure 7 and 32 bit is marked by orange outlined box in figure 8, for reference.

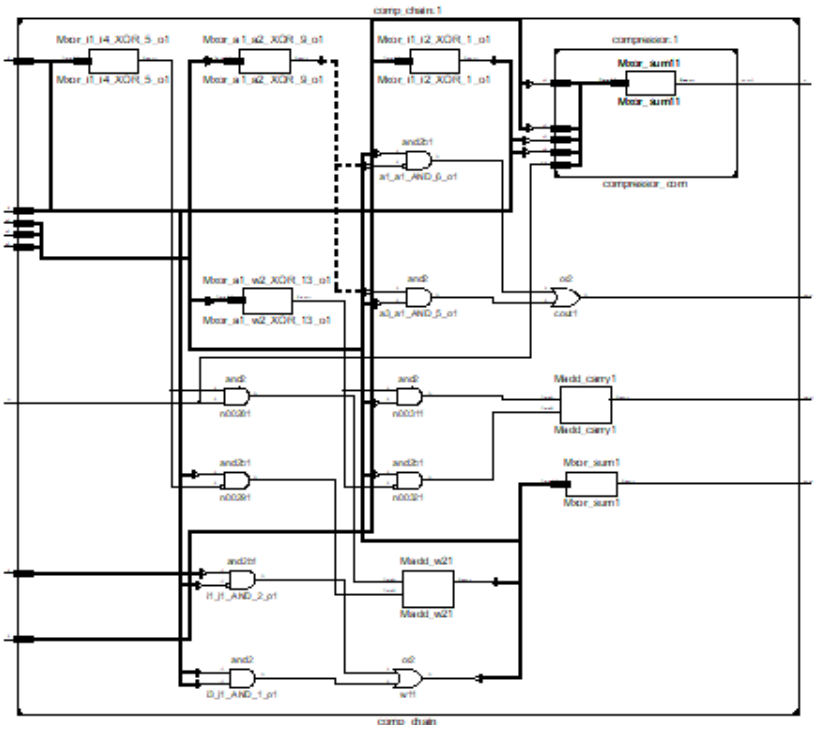

Fig 3: Gate level 8:4 compressor

Half Adder: A half adder produces two outputs bits: carry and sum, by adding two input bits. It is composed of an AND gate and an EXOR gate.

Table 2: Truth table of Half adder.

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

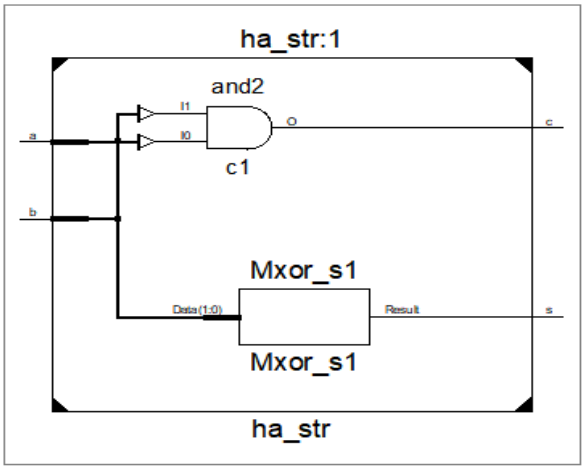

Fig 4: Half Adder

Approximate Full Adder: The adder is composed of three, two input AND gate, one three input OR gate and one NOT gate. This approximate full adder is exact same carry but two erroneous sum output, with compact design and along with a hint of less delay.

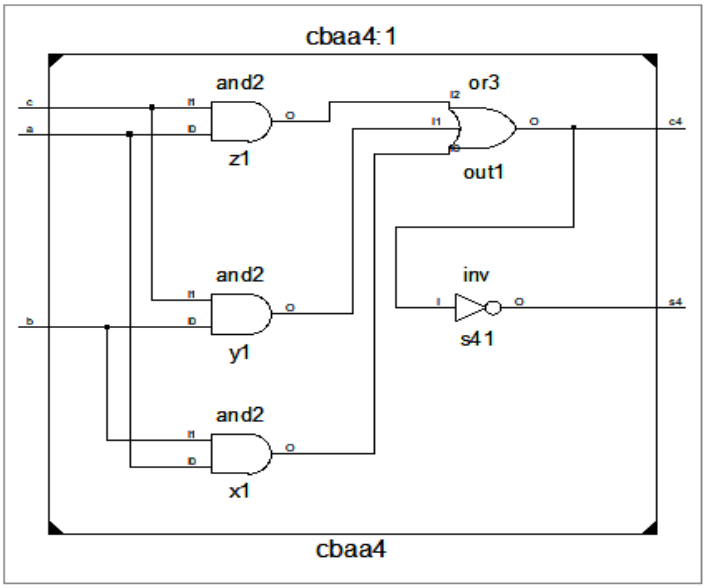

Fig 5: Approximate Full Adder

Table 3: Truth table of Approximate Full Adder.

A	B	C	Sum	Carry
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

3.2 MODIFIED APPROXIMATE MULTIPLIERS

Different multipliers are designed with different algorithms and they have different purpose according to their usage. Array multipliers and booth multipliers are slowest among all. Array multiplier lags due to its particular sequence in carry propagation through the array structure. Booth multiplier is also same as array multipliers. Wallace tree multiplier is power hungry model.

The Dadda multiplier, known for its efficient and high-speed multiplication capabilities, has attracted considerable interest due to its potential for approximation. Approximate Dadda multiplier designs, focus on their potential to reduce power consumption while maintaining an acceptable level of accuracy in computation results. The Dadda multiplier is a commonly used radix-4 multiplier that exhibits efficient partial product reduction, resulting in reduced hardware complexity and improved speed. It utilizes a tree-like structure to perform multiplication operations, reducing the number of partial products and intermediate adders required compared to traditional multiplication techniques. This inherent parallelism and optimized structure make the Dadda multiplier an ideal condition for approximation. Various approximation techniques employed in Dadda multipliers to achieve energy efficiency without compromising overall system accuracy. These techniques exploit the redundancies in the multiplication process and apply selective approximations to eliminate unnecessary operations. Common approaches include reducing the number of partial products, approximating partial products, employing truncated representations and adopting approximate adders. Approximate Dadda multipliers offer a promising solution for energy-efficient arithmetic computations.

The modified design of 8-bit, 16-bit and 32-bit approximate multiplier is given below. Multiplication operation is divided into three stages namely:

1) Generation of partial products.

2) Arrangement of partial products into compressors and bit wise adder’s division.

3) In final stage only half adders are applied to achieve the final output.

The unsigned 8bit, 16bit and 32 bit Dadda multiplier can be optimized by improving stage two arrangements. Each dot represents a product functions like (AND, OR, NOT, XOR, XNOR, NAND and NOR) present within the circuit. Approximate compressors are indicated by green box, full adders by red box and half adders by dotted box.

I have decided to implement this approximate full adder by replacing traditional full adder because it gives exact same carry but two erroneous sum output, with compact design and along with a hint of less delay.

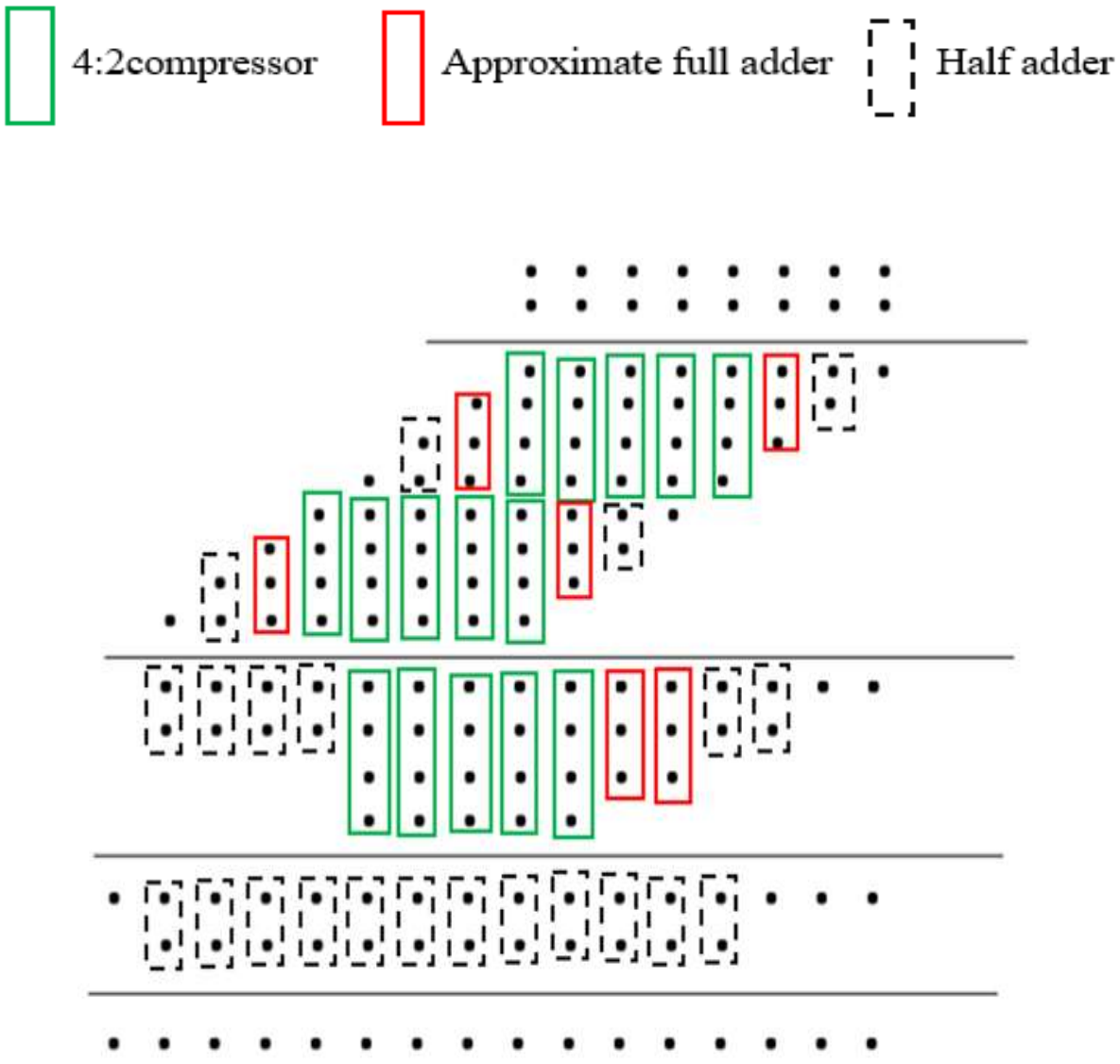

Fig 6: 8-bit Approximate Multiplier

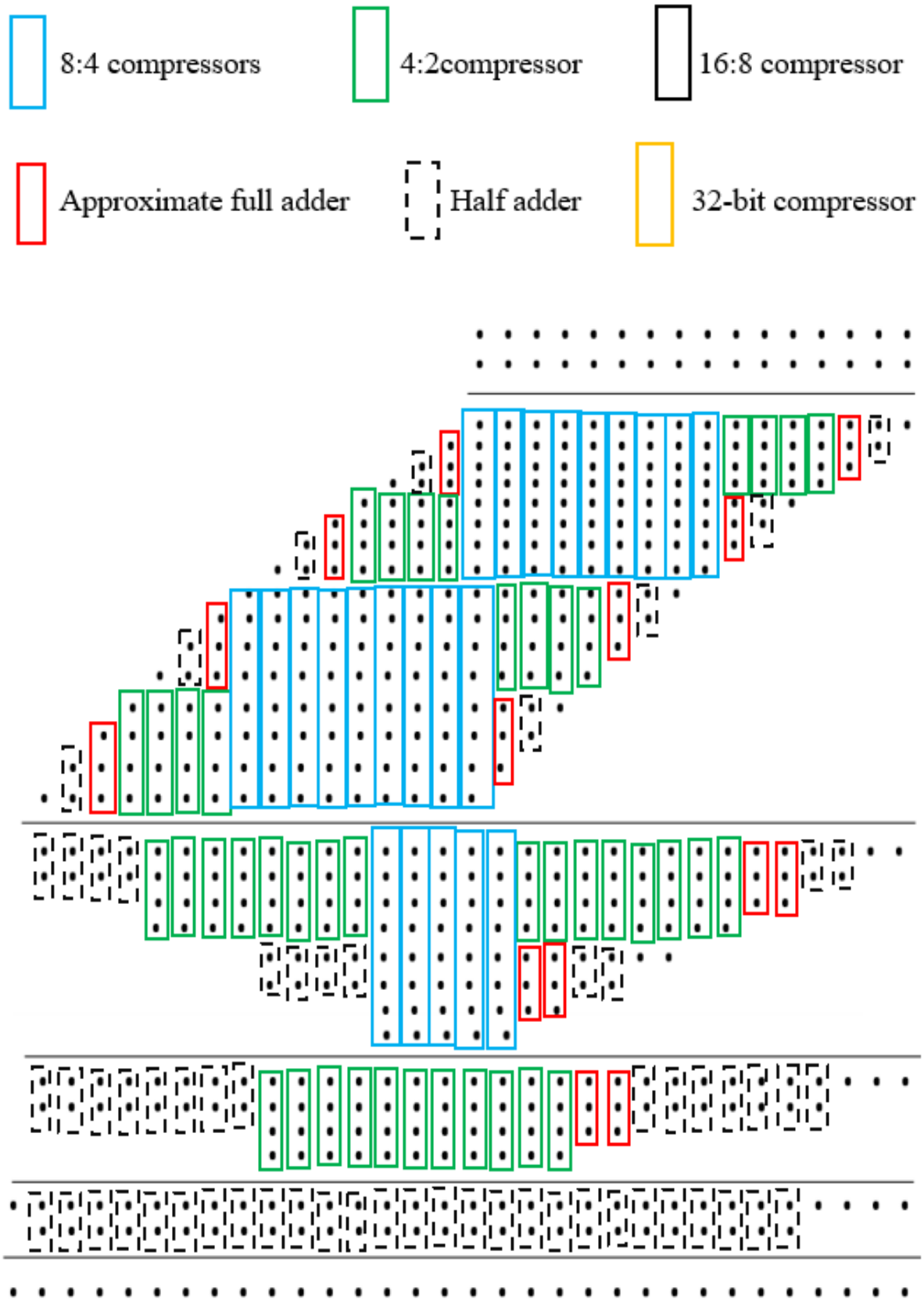

Fig 7: 16-bit Approximate Multiplier

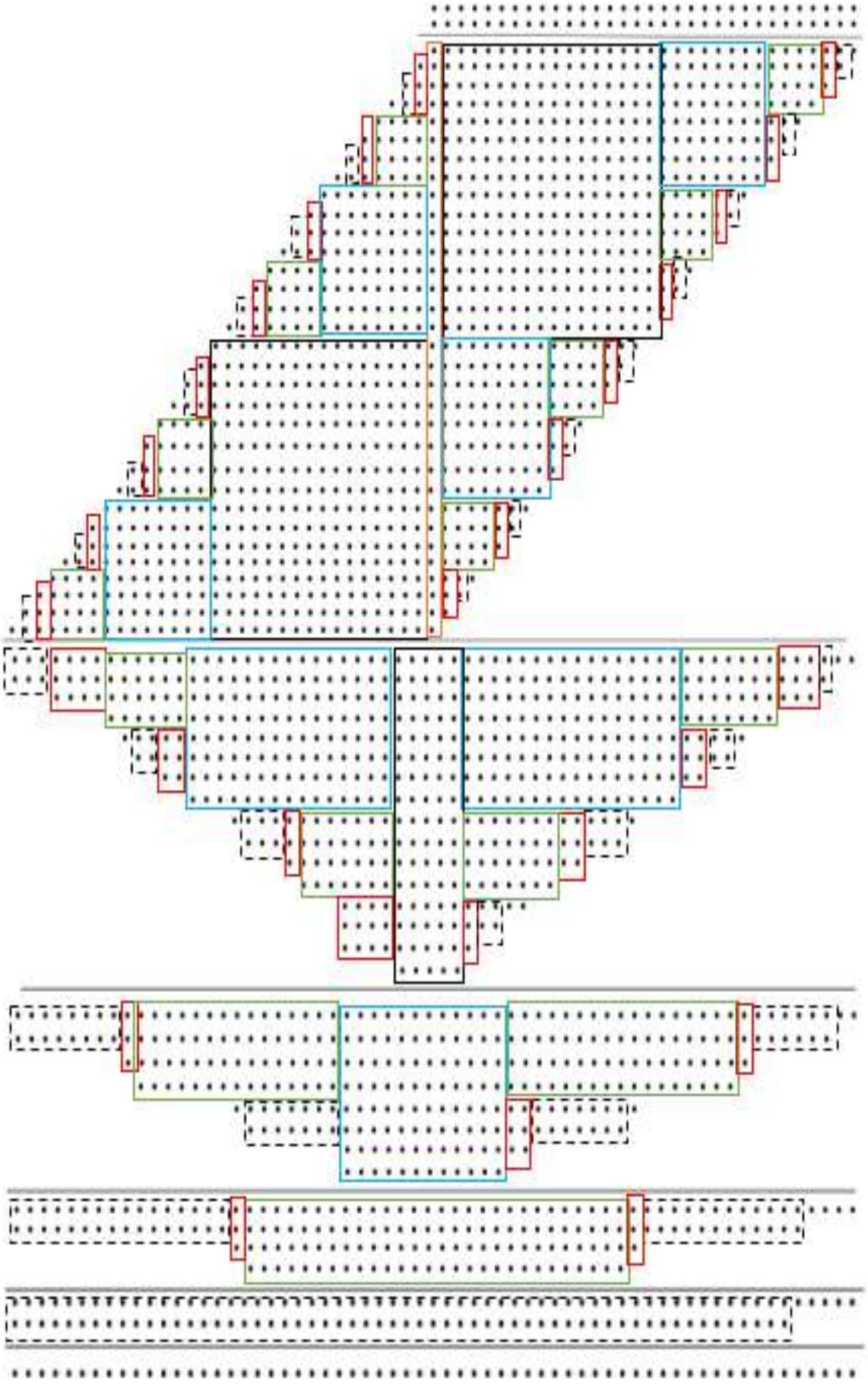

Fig 8: 32-bit Approximate Multiplier

XILINX:

A software called Xilinx Tools is used to design digital circuits that are implemented using Xilinx Field Programmable Gate Array (FPGA) or Complex Programmable Logic Device(CPLD) technology. The design process is composed of the following steps:

(a) Design algorithm and architecture.

(b) Design synthesis and implementation.

(c) Functional simulation.

(d) Testing and verification.

The aforementioned CAD tools allow for the entry of digital designs utilizing a variety of methods, including the use of a schematic entry tool, a hardware description language (HDL) such as Verilog or VHDL, or a mix of both. Verilog HDL will be used exclusively for the design flow.

Starting with Verilog HDL designing, the CAD tools create combinational and sequential circuits. The following is a list of the steps in this design process:

1. Use a template-driven editor to create the necessary Verilog design input files.

2. Compile the Verilog design files and implement them.

3. Without utilizing a PLD (FPGA or CPLD), create the test-vectors and simulate the design (functional simulation).

4. Assign input/output pins to a target device in order to implement the design.

5. Download the bit stream to a CPLD or FPGA.

6. CPLD/FPGA device test design.

The following segments make up a Verilog input file in the Xilinx software environment:

Header: Module name, a list of the ports for input and output.

Declaration: Ports for input and output, registers, and wires are declared.

Logic Descriptions: Equations, state machines and logic functions.

End: final module.

By replicating the HDL description, we may check design functionality early in the design flow. We can make any necessary modifications early on by testing design choices before the design is implemented at the gate or register transfer level. Hardware description to be implemented on a FPGA device:

• Design time will be reduced by using a higher-level design specification rather than expressing the design from the FPGA device base elements when you synthesize hardware description to target the FPGA device implementation.

• Lowers the possibility of human error when converting a hardware description into a schematic design.

• Enables the original Hardware Description Language (HDL) code to be modified using the automation techniques employed by the synthesis tool (such as machine encoding styles and automatic I/O insertion) during optimization. Greater optimization and efficiency follow from this.

3.3 PERFORMANCE ANALYSIS:

To assess the performance of approximate Dadda multipliers, a comprehensive set of evaluation metrics is presented in terms of area (LUTs), delay, power, PDP, PLP and PLDP. The impact of approximation techniques on these metrics are discussed, and modified design with valuable insights for selective appropriate techniques are based on specific application requirements.

Table 4: Comparison of Area (LUTs), Delay and Power consumption of existing and modified 8, 16 and 32bit Approximate Multipliers

Device Family	Data Bit	Scheme	Area (LUTs)	Delay (ns)	Power (W)	PDP (W.ns)	PLP (W)	PLDP (W.ns)
Artix-7 (Basys-3)	8	Evadoor et al.	70	2.501	0.862	2.1559	60.34	150.91034
		Modified	67	2.462	0.84	2.0681	56.28	138.56136
	16	Evadoor et al.	457	4.384	1.842	8.0753	841.8	3690.4249
		Modified	438	4.142	1.738	7.1988	761.2	3153.0726
	32	Evadoor et al.	2168	6.035	2.462	14.858	5338	32212.513
		Modified	2082	5.355	2.388	12.788	4972	26624.075

Approximate Dadda multiplier designs from the literature, discussing their strengths and weaknesses. Experimental results and case studies demonstrate the efficacy of the designs in terms of energy efficiency and computational accuracy. Furthermore, comparisons with traditional multiplier designs and other approximate multipliers are provided, highlighting the advantages of the approximate Dadda multiplier.

Table 5: Comparison of 8 and 16 bit existing and modified Multipliers in terms of Area, Delay and Power.

Device Family	Data Bit	Multiplier	Area (LUTs)	Delay (ns)	Power (W)	PDP (W.ns)	PLP (W)	PLDP (W.ns)
Artix-7 (Basys-3)	8	Array	109	6.435	1.108	7.13	120.8	777.16782
		Booth	135	8.99	1.124	10.105	151.7	1364.1426
	16	Array	266	9.052	1.989	18.004	529.1	4789.1778
		Modified Booth	252	8.438	1.871	15.787	471.5	3978.4495

From Table 4 we can conclude that modified multiplier is showing good aspect when compared to state of art multipliers. Table 5 helps in understanding use of Dadda multiplier. When checking PDP, PLP and PLDP we can see that there is a huge improvement compared to Edavoor et al. PDLP is observed and we can see significant improvement. Table 4 and Table 5 when compared for 8 and 16 bit we can see the major difference is occurring in PLDP relatively. Delay is significantly increased in 32 bit compared to 8 and 16 bit. Power consumption of 16bit is comparatively same in both Table 4 and 5, when area (LUTs) are halved in 16 bit in Table 5 compared to Table 4. From Table 4 we can conclude that the bit and approximation is proportional positively.

This work provides a comprehensive overview of the existing approximation techniques employed in Dadda multipliers, along with their evaluation metrics and performance analysis.

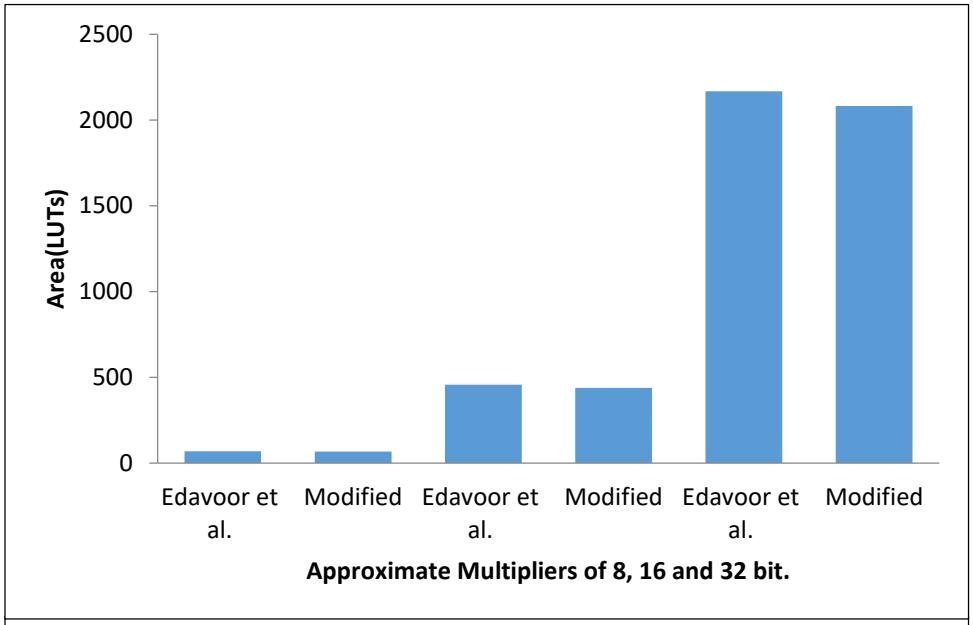

Fig 9: Area comparison of 8, 16 and 32bit Approximate Multiplier

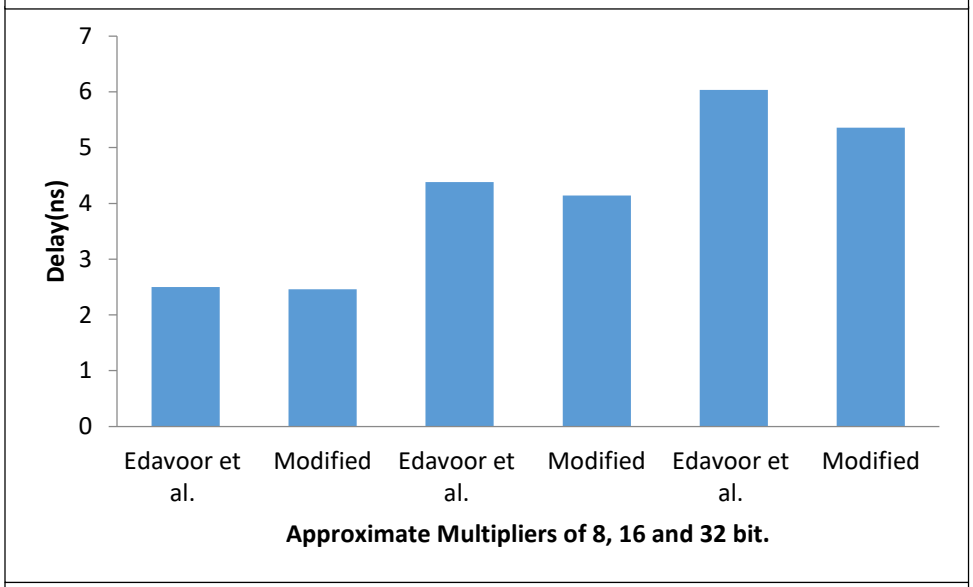

Fig 10: Delay comparison of 8, 16 and 32bit Approximate Multiplier

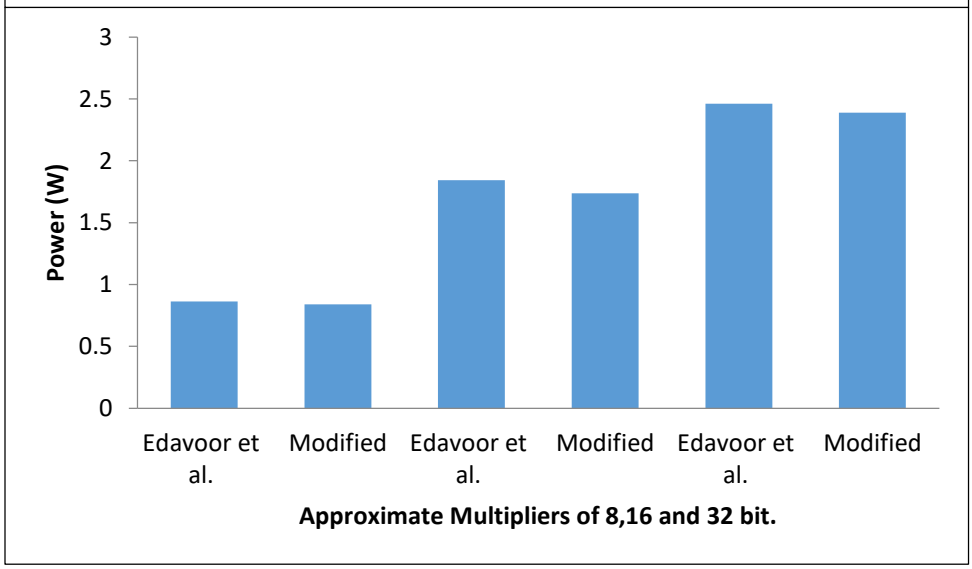

Fig 11: Power comparison of 8, 16 and 32bit Approximate Multiplier

The modified approximate multiplier containing exact compressors and approximate adders are stimulated and synthesized by Xilinx 14.7, from above comparison table it can be deduced that 3% more efficient in both area LUTs and power consumption. Modified design also have significant improvement in delay.

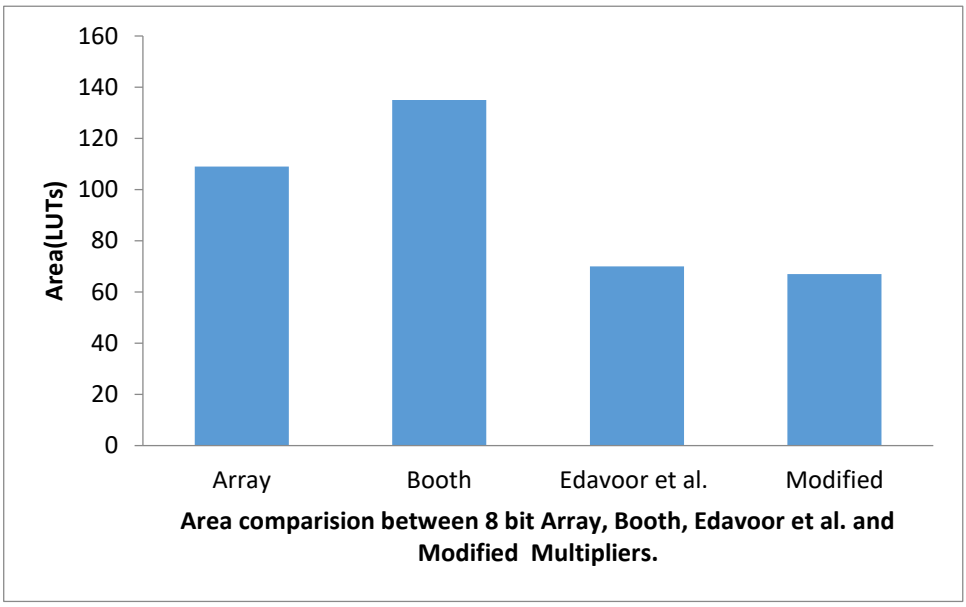

Fig 12: Area comparison between 8-bit Array, Booth, Edavoor et al. and Modified Approximate Multiplier

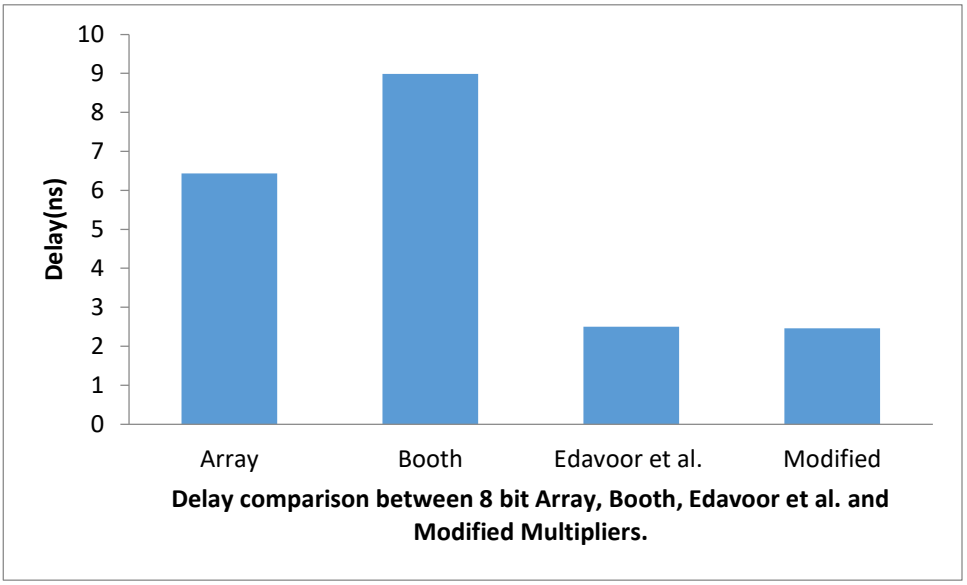

Fig 13: Delay comparison between 8-bit Array, Booth, Edavoor et al. and Modified Approximate Multiplier

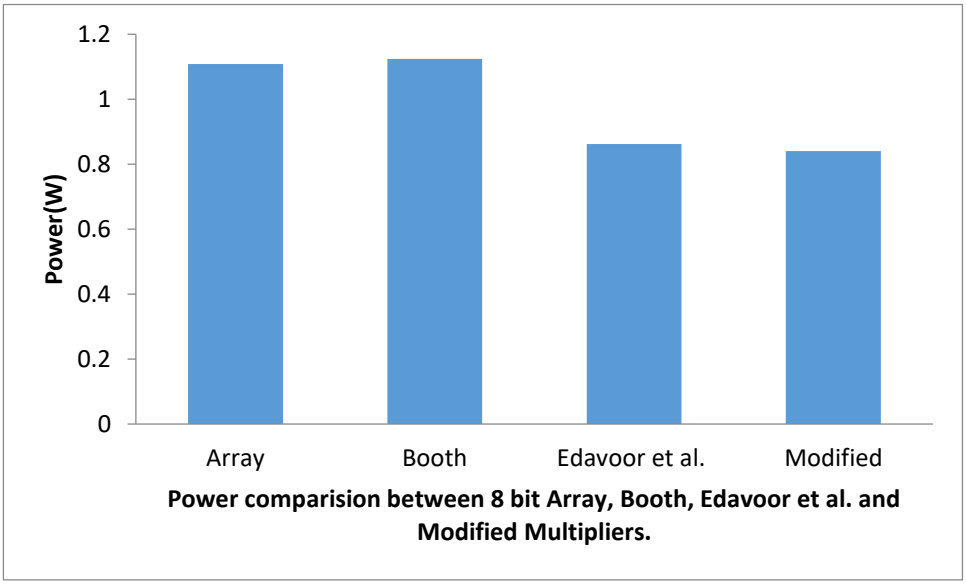

Fig 14: Power comparison between 8-bit Array, Booth, Edavoor et al. and Modified Approximate Multiplier

From the above compression of various approximate multipliers, it is observed that Dadda multipliers (Edavoor et al and modified) are advantageous than Array and Booth multipliers in area, delay and power. Among the Dadda multipliers, modified multiplier is improved version of Edavoor et al. multiplier.

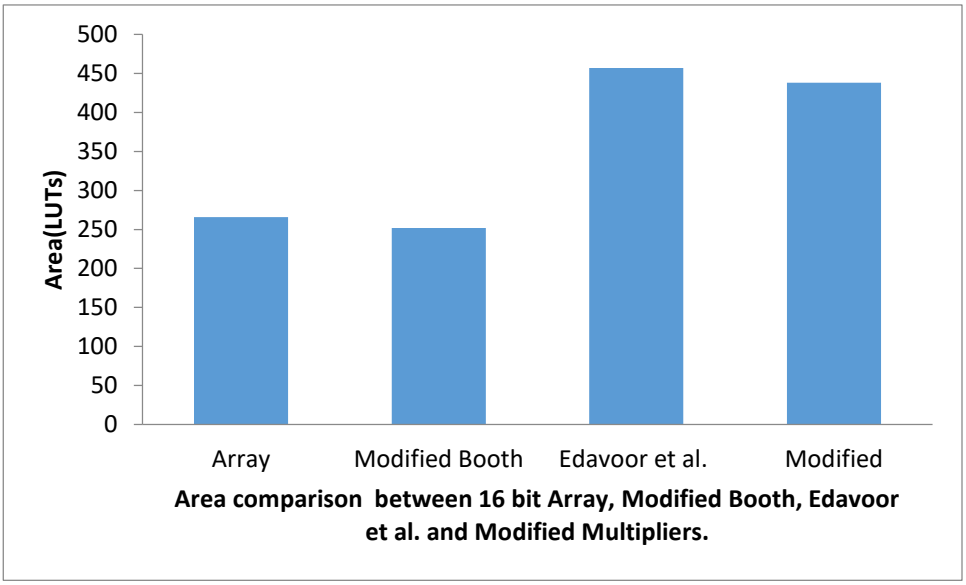

Fig 15: Area comparison between 16-bit Array, Booth, Edavoor et al. and Modified Approximate Multiplier

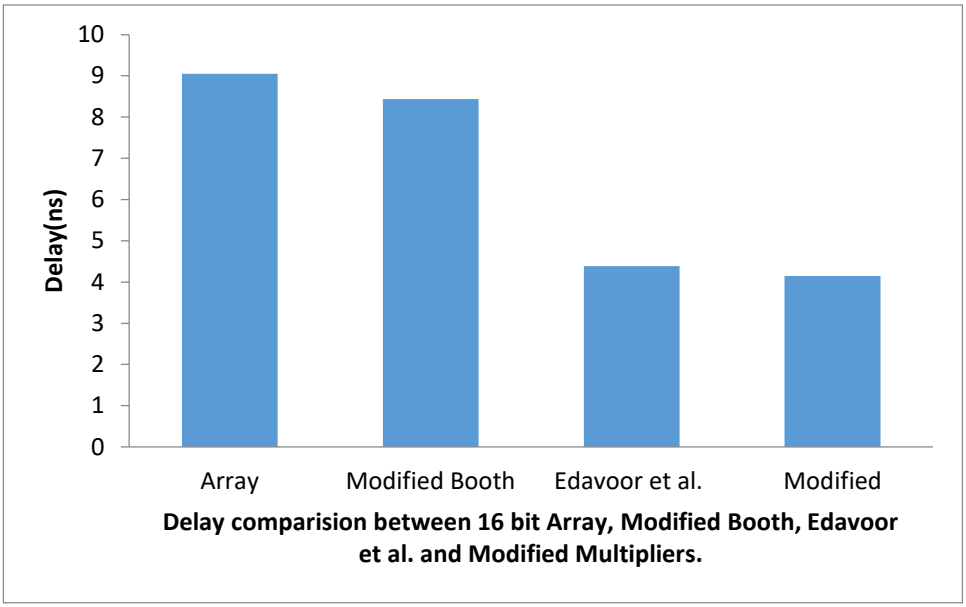

Fig 16: Power comparison between 16-bit Array, Booth, Edavoor et al. and Proposed Multiplier

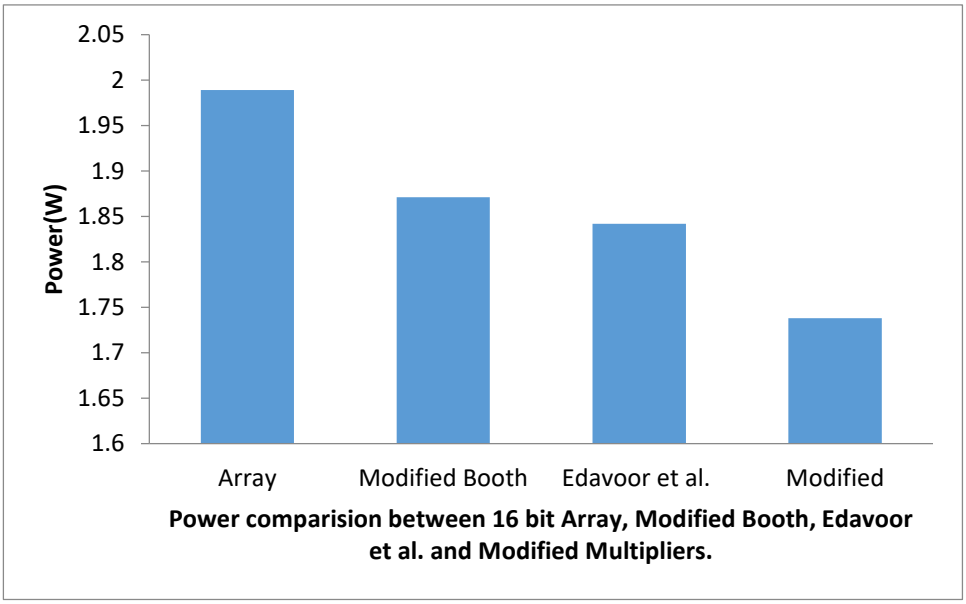

Fig 17: Delay comparison between 16-bit Array, Booth, Edavoor et al. and Proposed Multiplier

From comparison of various multipliers, it is observed that Dadda multipliers (Edavoor et al. and modified) are consuming more area relatively. Despite that both the Dadda multipliers show observable improvement in both dealay and power consumption.

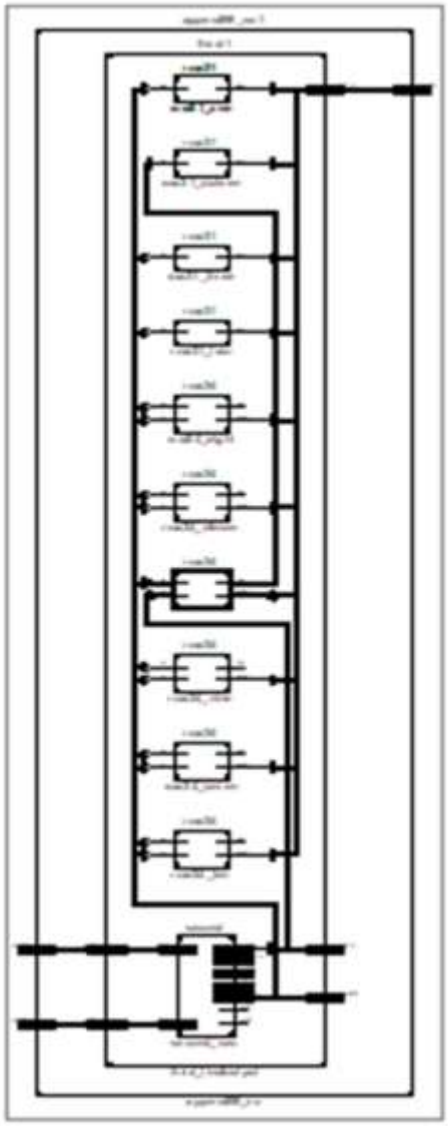

Fig 18: RTL Diagram: 8-bit Approximate Multiplier

(Modified)

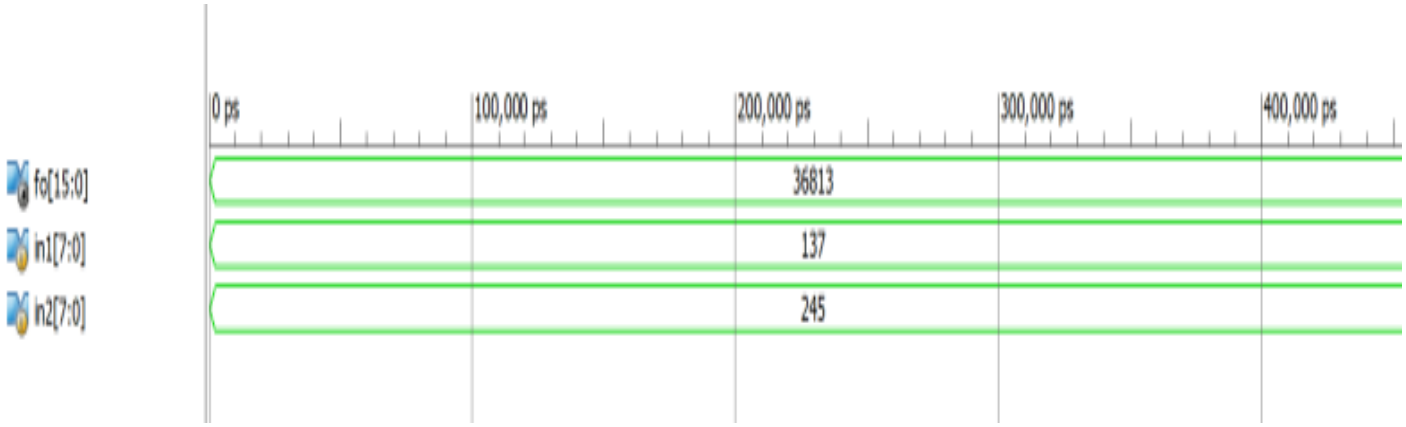

Fig 19: Waveform: 8-bit Approximate Multiplier (Modified)

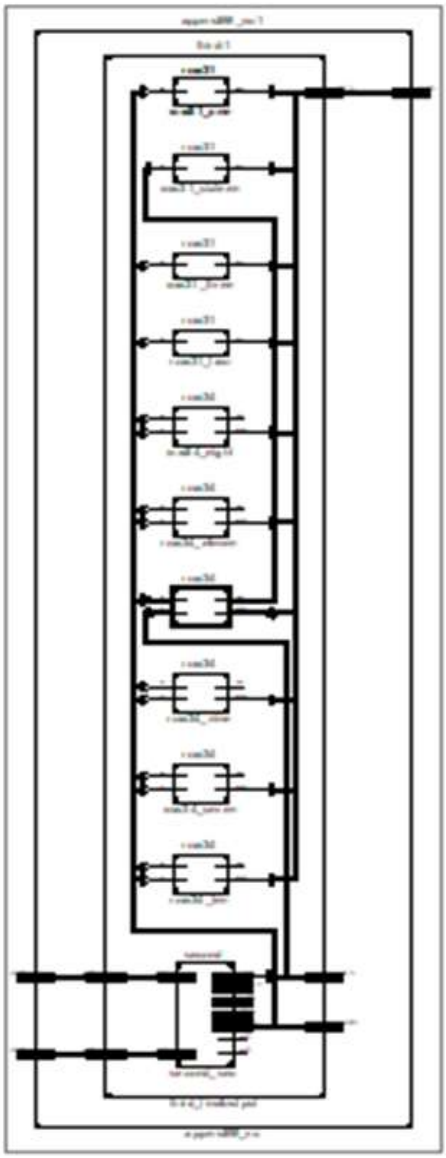

Fig 20: RTL Diagram: 8-bit Approximate Multiplier

(Edavoor et al.)

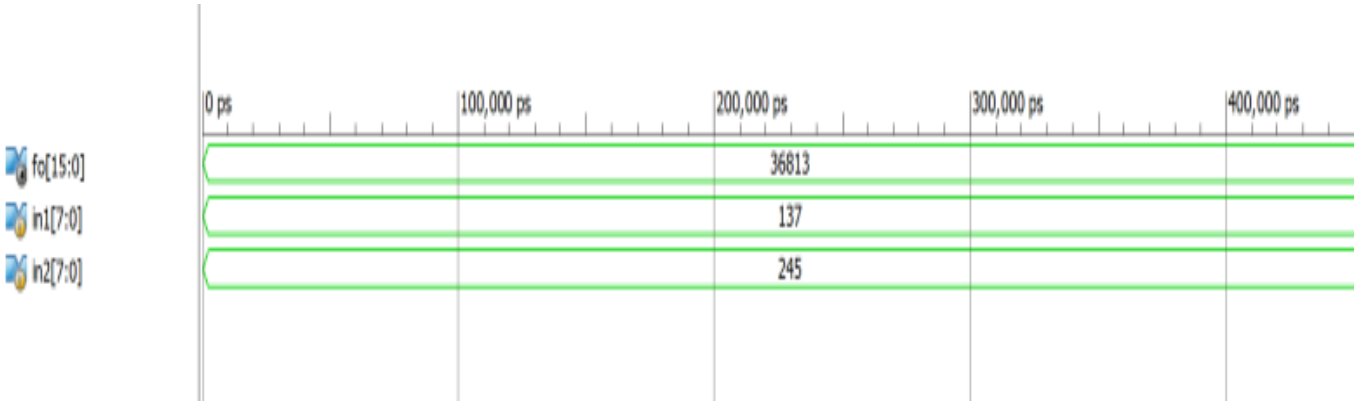

Fig 21: Waveform: 8-bit Approximate Multiplier (Edavoor et al.)

Fig 22: Waveform: 16-bit Approximate Multiplier (Modified)

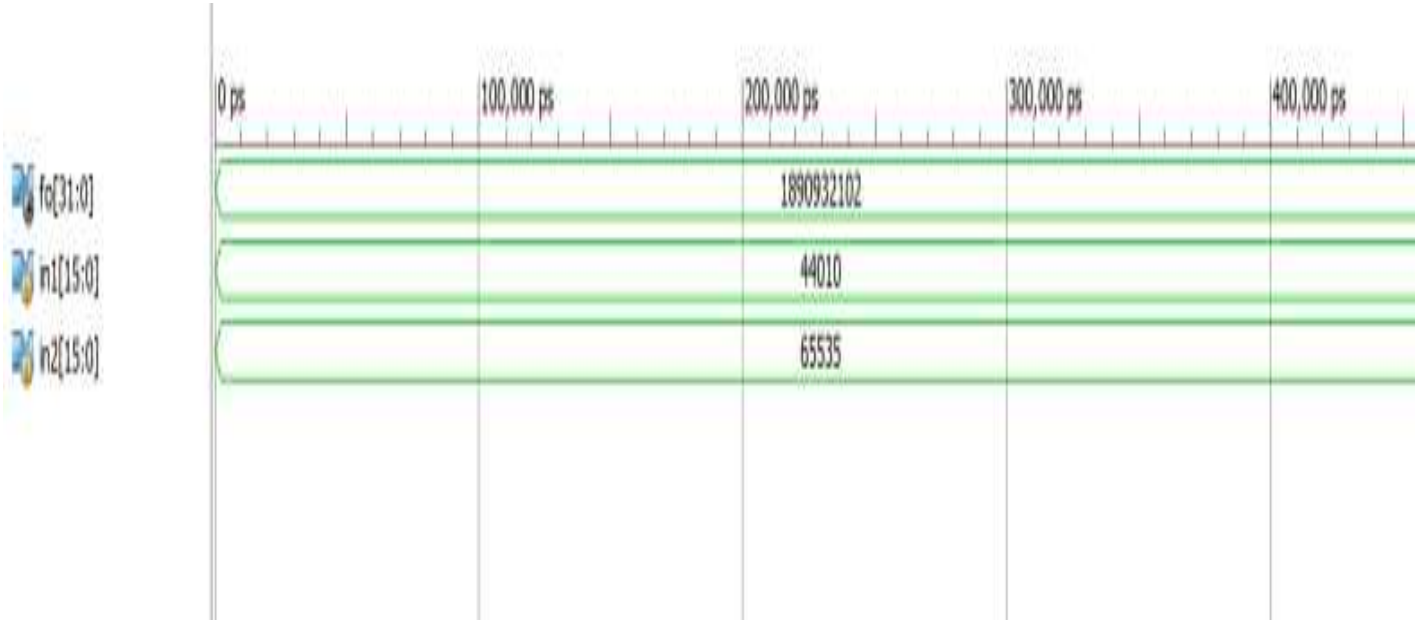

Fig 23: Waveform: 16-bit Approximate Multiplier (Edavoor et al.)

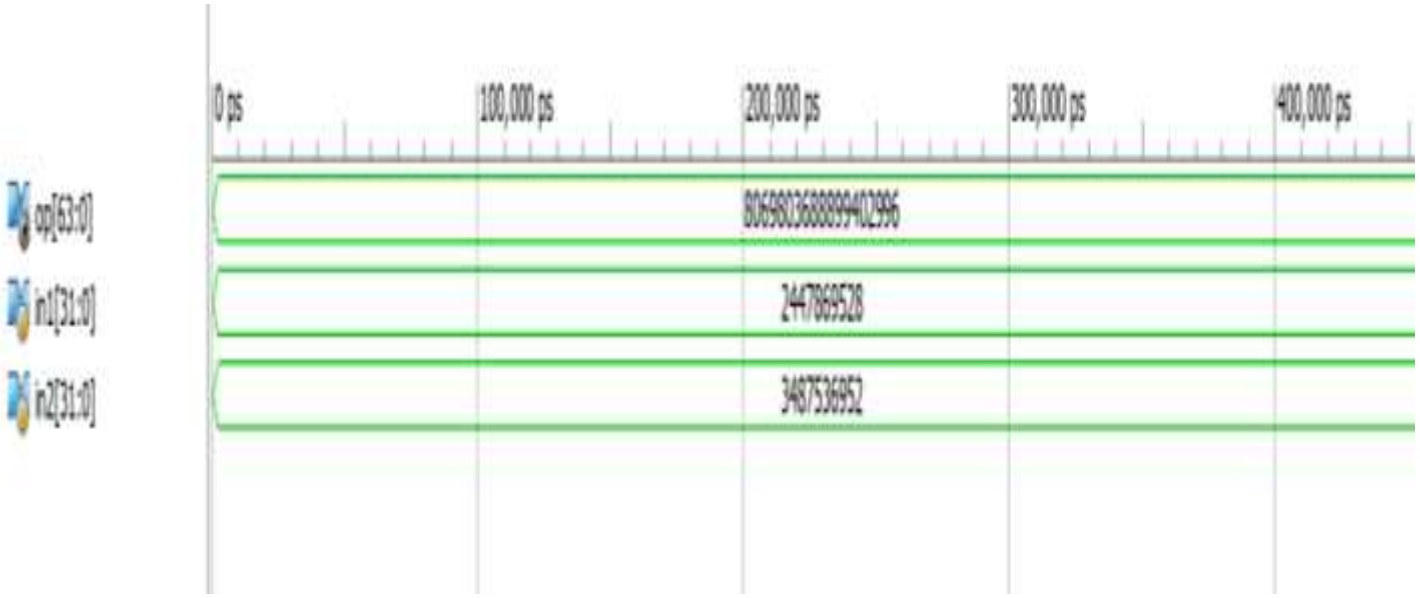

Fig 24: Waveform: 32-bit Approximate Multiplier (Modified)

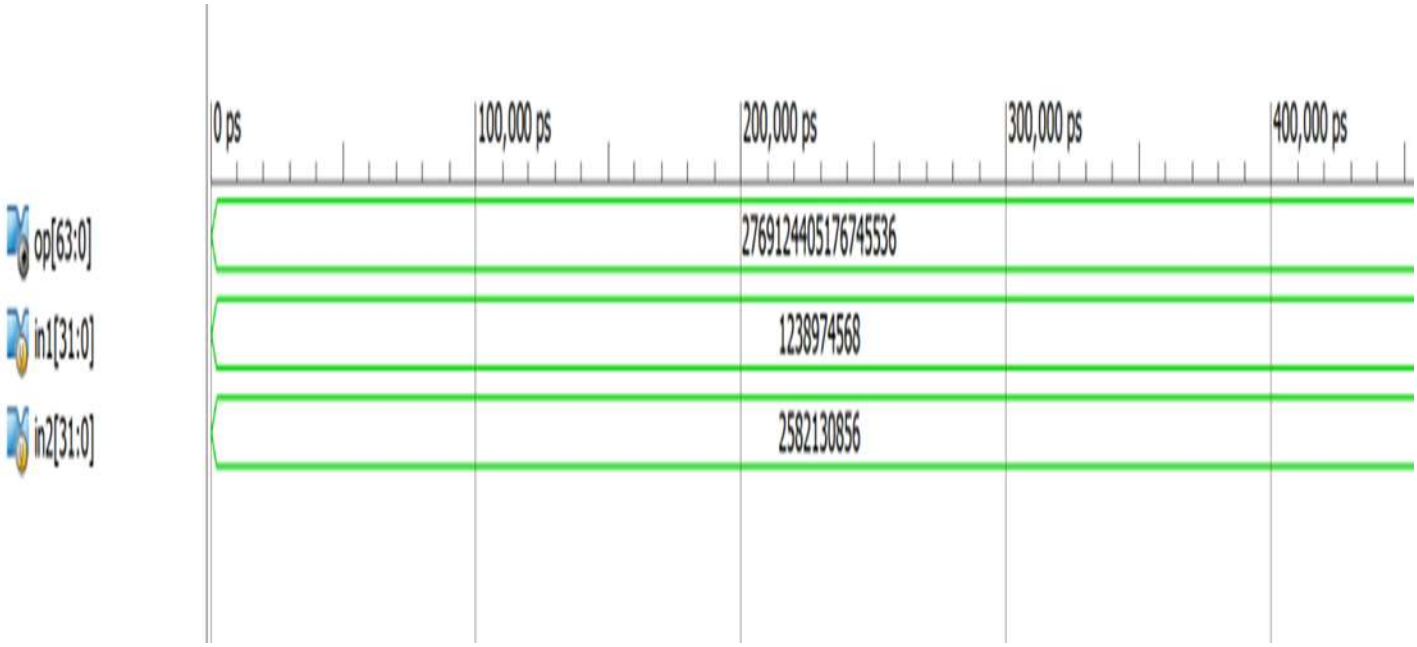

Fig 25: Waveform: 32-bit Approximate Multiplier (Edavoor et al.)

CHAPTER 5

CONCLUSION
&
FUTURE SCOPE

CONCLUSION

The use of approximate multipliers in VLSI designs accepts an amount of error and avoiding its spread across the system are two of the fundamental issues. The selected approximation technique is to offer a workable option for high-performance and power-efficient computation. Their capacity to strike a balance between accuracy and computational expense offers an alluring opportunity for a variety of applications. The design and error control tactics depending on the unique requirements of the target application in order to fully realize the potential of approximation multipliers.

Future research directions can include the exploration of novel approximation techniques, optimization algorithms, and architectural enhancements to further improve the energy efficiency and accuracy trade-offs in Dadda multipliers.

REFERENCE

1. P. J. Edavoor, S. Raveendran, and A. D. Rahulkar, “Approximate multiplier design using novel dual-stage 4: 2 compressors.” IEEE Access 8, pp. 48337-48351, Mar 2020.

2. S. Hashemi, “Approximate Computing Techniques for Accuracy-Energy Trade-offs.” Doctoral dissertation, Brown University Providence, Rhode Island, 2018.

3. A. Raha, S. Venkataramani, V. Raghunathan, and A. Raghunathan, "Quality configurable reduce-and-rank for energy efficient approximate computing." Design, Automation & Test in Europe Conference & Exhibition, pp. 665-670. IEEE, Mar 2015.

4. N. Arya, and M. Rahul, "A new trend in VLSI based multipliers for error resilient DSP applications." Int Res J Eng Technol (IRJET) 5, no. 4, pp. 3866-3869, 2018.

5. M. Ramasamy, G. Narmadha, and S. Deivasigamani, "Carry based approximate full adder for low power approximate computing." 7th International Conference on Smart Computing & Communications (ICSCC), pp. 1-4. IEEE, Jun 2019.

6. H. Jiang, F. J. H. Santiago, H. Mo, L. Liu, and J. Han, "Approximate arithmetic circuits: A survey, characterization, and recent applications." Proceedings of the IEEE 108, no. 12, pp. 2108-2135, Aug 2020.

7. L. M. Adams, "Improving the Hardware Performance of Arithmetic Circuits using Approximate Computing." PhD diss., University of Saskatchewan, 2020.

8. D. Catelan, R. Santos, and L. Duenha, "Accuracy and physical characterization of approximate arithmetic circuits." Anais do XXI Simpósio em Sistemas Computacionais de Alto Desempenho, pp. 143-154. SBC, Oct 2020.

9. C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery." Design, Automation & Test in Europe Conference & Exhibition, pp. 1-4. IEEE, Mar 2014.

10. A. Agrawal, J. Choi, K. Gopalakrishnan, S. Gupta, R. Nair, J. Oh, D. A. Prener, S. Shukla, V. Srinivasan, and Z. Sura, "Approximate computing: Challenges and opportunities." IEEE International Conference on Rebooting Computing (ICRC), pp. 1-8. IEEE, Oct 2016.

11. S. Muthulakshmi, C. S. Dash, S.R.S. Prabaharan, “Memristor augmented approximate adders and subtractors for image processing applications: An approach,” AEU - International Journal of Electronics and Communications, Volume 91, 2018, Pages 91-102, ISSN 1434-8411, Jul 2018.

12. G. Anusha, and P. Deepa. "Design of approximate adders and multipliers for error tolerant image processing." Microprocessors and Microsystems 72: 102940, Feb 2020.

13. Y. Saraswati, M. Manjula, N. Divya and D. Sirisha, “Implementation of Approximate Dadda Multiplier with inexact compressor” Journal of Engineering Science, Vol 10, Issue 12, ISSN NO:0377-9254, Dec 2019.

14. K. M. Reddy, M. H. Vasantha, Y. N. Kumar, and D. Dwivedi. "Design and analysis of multiplier using approximate 4-2 compressor." AEU-International Journal of Electronics and Communications 107: 89-97, Jul 2019.

15. M. S. Ansari, "Hardware-Efficient Approximate Arithmetic Circuits for Deep Learning and Other Computation-Intensive Applications." (2020).

16. M. H. Moaiyeri, F. Sabetzadeh, and S. Angizi. "An efficient majority-based compressor for approximate computing in the nano era." Microsystem Technologies 24: 1589-1601, Mar 2018.

17. L. Maddisetti, and J.V.R. Ravindra, “Machine learning based power efficient approximate 4: 2 compressors for imprecise multipliers.” 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), pp. 221-226, IEEE, Jan 2019.

18. K. C. Pathak, J. N. Sarvaiya, A. D. Darji, S. Diwan, A. Gangadwala, Z. Bhatt, and A. Patel, "An Efficient Dadda Multiplier using Approximate

Adder." IEEE Region 10 Conference (TENCON), pp. 176-181. IEEE, Nov 2020.

19. S. Chandaka, and B. Narayanam, "Hardware Efficient Approximate Multiplier Architecture for Image Processing Applications." Journal of Electronic Testing 38, no. 2: 217-230, Apr 2022.

20. F. Ranjbar, Y. Forghani, and D. Bahrepour, "High performance 8-bit approximate multiplier using novel 4: 2 approximate compressors for fast image processing." International Journal of Integrated Engineering 10, no. 1, Apr 2018.

21. S. Sudha and D. Nirosha, “Design of Dadda Multiplier for Image Applications” International Journal of Advanced Research Trends in Engineering and Technology (IJARTET) Vol. 3, Special Issue 15, Mar 2016.

22. J. Ramasamy and S. Nagarajan, "Hybrid segment approximate multiplication for image processing applications." Circuits and Systems 7, no. 08: 1701, 2016.

23. B. H. Rao, and V. R. Kumar, "Implementation of 8x8 Dadda Multiplier using approximate compression for image enhancement." (IJAER) 10, no. 1, Jul 2015.

24. C. S. R. Reddy, U. A. Kumar and S. E. Ahmed, "Design of efficient approximate multiplier for image processing applications." Modelling,

Simulation and Intelligent Computing: Proceedings of MoSICom 2020, pp. 511-518. Springer Singapore, 2020.

25. S. P. J. V. Rani, J. R. L. Jennifer and P. Sudhanya. "Approximate Multipliers Design Using Approximate Adders for Image Processing Applications." Journal of Circuits, Systems and Computers 31.15: 2250256, Oct 2022.

26. L. H. Krishna, J. B. Rao, S. Ayesha, S. Veeramachaneni and S.N Mahammad, “Energy Efficient Approximate Multiplier Design for Image/Video Processing Applications.” IEEE International Symposium on Smart Electronic Systems (iSES) pp. 210-215, IEEE. Dec 2021.

27. Y. Zhao, T. Li, F. Dong, Q. Wang, W. He and J. Jiang, "A new approximate multiplier design for digital signal processing." IEEE 13th international conference on ASIC (ASICON), pp. 1-4. IEEE, Oct 2019.

28. C. V. Gowdar and M. C. Parameshwara, "Design of Energy Efficient Approximate Multipliers for Image Processing Applications." ICTACT Journal on Microelectronics 7, no. 1: 1057-1061, 2021.

29. K. M. Reddy, M. H. Vasantha, Y. N. Kumar and D. Dwivedi, "Design and analysis of multiplier using approximate 4-2 compressor." AEU-International Journal of Electronics and Communications 107: 89-97, Jul 2019.

30. A. Gorantla, and P. Deepa, "Design of approximate compressors for multiplication." ACM Journal on Emerging Technologies in Computing Systems (JETC) 13, no. 3: 1-17, 2017.

31. Y. Guo, H. Sun, L. Guo and S. Kimura, "Low-cost approximate multiplier design using probability-driven inexact compressors." IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 291-294. IEEE, Oct 2018.

32. R. Marimuthu, Y. E. Rezinold and P. S. Mallick, "Design and analysis of multiplier using approximate 15-4 compressor." IEEE Access 5: 1027-1036, Dec 2016.

33. H. Pei, X. Yi, H. Zhou and Y. He, "Design of ultra-low power consumption approximate 4–2 compressors based on the compensation characteristic." IEEE Transactions on Circuits and Systems II: Express Briefs 68, no. 1: 461-465, Jun 2020.

34. S. Vahdat, M. Kamal, A. Afzali-Kusha and M. Pedram. "TOSAM: An energy-efficient truncation-and rounding-based scalable approximate multiplier." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27, no. 5: 1161-1173, Jan 2019.

35. J. Lee, H. Seo, Y. Kim and Y. Kim, "Approximate adder design with simplified lower-part approximation." IEICE Electronics Express 17, no. 15: 20200218-20200218, Aug 2020.

36. T. Yang, T. Ukezono and T. Sato, "A low-power configurable adder for approximate applications." 19th International Symposium on Quality Electronic Design (ISQED), pp. 347-352. IEEE, Mar 2018.

37. R. Nayar, P. Balasubramanian and D. L. Maskell, "Hardware Optimized Approximate Adder with Normal Error Distribution," IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Limassol, Cyprus, 2020, pp. 84-89, Jul 2020.

38. B. S. Prabakaran, "DeMAS: An efficient design methodology for building approximate adders for FPGA-based systems," Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 2018, pp. 917-920, Mar 2018.

39. K. Al-Maaitah, G. Tarawneh, A. Soltan, I. Qiqieh and A. Yakovlev, "Approximate adder segmentation technique and significance-driven error correction," 2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), Thessaloniki, Greece, 2017, pp. 1-6, Sep 2017.

40. H. Seo, Y. S. Yang and Y. Kim. "Design and analysis of an approximate adder with hybrid error reduction." Electronics 9, no. 3 (2020): 471, Mar 2020.

41. A. G. M. Strollo, D. De Caro, E. Napoli, N. Petra and G. Di Meo, "Low-Power Approximate Multiplier with Error Recovery using a New

Approximate 4-2 Compressor," IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 2020, pp. 1-4, Oct 2020.

42. H. Pei, X. Yi, H. Zhou and Y. He, "Design of Ultra-Low Power Consumption Approximate 4–2 Compressors Based on the Compensation Characteristic," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 1, pp. 461-465, Jan. 2021.

43. D. Esposito, A. G. M. Strollo, E. Napoli, D. De Caro and N. Petra, "Approximate Multipliers Based on New Approximate Compressors," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4169-4182, Dec. 2018.

44. M. Ha and S. Lee, "Multipliers With Approximate 4–2 Compressors and Error Recovery Modules," in IEEE Embedded Systems Letters, vol. 10, no. 1, pp. 6-9, March 2018.

45. M. Ahmadinejad and M. H. Moaiyeri, "Energy- and Quality-Efficient Approximate Multipliers for Neural Network and Image Processing Applications," in IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 2, pp. 1105-1116, Apr 2022.

46. C. H. Lin and I. C. Lin, "High accuracy approximate multiplier with error correction," 2013 IEEE 31st International Conference on Computer Design (ICCD), Asheville, NC, USA, pp. 33-38, Oct 2013.

47. V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, "Low-Power Digital Signal Processing Using Approximate Adders," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124-137, Jan. 2013.

48. M. S. Ahmed, "Approximate Computing Architectures and Algorithms for Error Resilient Applications." PhD diss., International Institute of Information Technology Hyderabad, 2021.

49. D. Catelan, R. Santos and L. Duenha, "Evaluation and characterization of approximate arithmetic circuits." Concurrency and Computation: Practice and Experience (2022): e6865, Feb 2022.

50. M. A. Hanif, V. Mrazek and M. Shafique. "Approximate Computing Architectures." In Handbook of Computer Architecture, pp. 1-41. Singapore: Springer Nature Singapore, Jan 2022.

51. D. Pandey, V. Mishra, S. Singh, S. Satapathy, B. Jajodia and D. S. Banerjee, "HPAM: An 8-bit High-Performance Approximate Multiplier Design for Error Resilient Applications," 23rd International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, pp. 1-5, Apr 2022.

52. O. Akbari, M. Kamal, A. Afzali-Kusha and M. Pedram, "Dual-Quality 4:2 Compressors for Utilizing in Dynamic Accuracy Configurable Multipliers," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 4, pp. 1352-1361, Apr 2017.

53. X. Yi, H. Pei, Z. Zhang, H. Zhou and Y. He, "Design of an Energy-Efficient Approximate Compressor for Error-Resilient Multiplications," IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, pp. 1-5, May 2019.

54. H. Afzali-Kusha, M. Vaeztourshizi, M. Kamal and M. Pedram, "Design Exploration of Energy-Efficient Accuracy-Configurable Dadda Multipliers With Improved Lifetime Based on Voltage Overscaling," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 5, pp. 1207-1220, May 2020.

55. A. G. M. Strollo, E. Napoli, D. De Caro, N. Petra and G. D. Meo, "Comparison and Extension of Approximate 4-2 Compressors for Low-Power Approximate Multipliers," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 9, pp. 3021-3034, Sept. 2020.

56. M. S. Ansari, V. Mrazek, B. F. Cockburn, L. Sekanina, Z. Vasicek and J. Han, "Improving the Accuracy and Hardware Efficiency of Neural Networks Using Approximate Multipliers," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 2, pp. 317-328, Feb. 2020.

57. S. Vahdat, M. Kamal, A. Afzali-Kusha and M. Pedram, "TOSAM: An Energy-Efficient Truncation- and Rounding-Based Scalable Approximate Multiplier," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 5, pp. 1161-1173, May 2019.

58. A. Towhidy, R. Omidi and K. Mohammadi, "On the Design of Iterative Approximate Floating-Point Multipliers," in IEEE Transactions on Computers, vol. 72, no. 6, pp. 1623-1635, 1 June 2023.

59. S. Zhang, Y. Wang, X. Chen, Y. Han, Y. Wang and X. Li, “Thread: Towards fine-grained precision reconfiguration in variable-precision neural network accelerator.” IEICE Electronics Express, 16(14), pp.20190145-20190145, 2019.

60. N. C. Huang, S. Y. Chen and K. C. Wu, "Sensor-Based Approximate Adder Design for Accelerating Error-Tolerant and Deep-Learning Applications," Design, Automation & Test in Europe Conference & Exhibition, Florence, Italy, pp. 692-697, Mar 2019.

61. M. Bruestel and A. Kumar, "Accounting for systematic errors in approximate computing," Design, Automation & Test in Europe Conference & Exhibition, Lausanne, Switzerland, 2017, pp. 298-301, Mar 2017.