doi.org/10.54473/IJTRET.2023.7201
International Journal of Trendy Research in Engineering and Technology
Volume 7 Issue 2 April 2023
ISSN NO 2582-0958

DESIGN AND IMPLEMENTATION OF A NOVEL ENERGY-EFFICIENT MULTIPLIER CIRCUIT USING APPROXIMATE COMPUTING TECHNIQUE

Visanthi. V.P, Assistant Professor, Department of Electronics,
College of Engineering, Cherthala, Kerala. visanthivpillai@gmail.com

Received 13 Feb 2023 Received in revised form 15 Feb 2023 Accepted 18 Feb 2023

ABSTRACT

A promising paradigm for many imprecision-tolerant applications has recently been identified as approximate arithmetic. By reducing accuracy standards, it can significantly reduce circuit complexity, latency, and energy consumption. In this research, we present a unique significance-driven logic compression (SDLC) approach for an energy-efficient approximate multiplier design. An algorithmic and adjustable lossy compression of the partial product rows based on their progressive bit importance is the core of this approach. To lessen the number of product rows, the resulting product terms are then commutatively remapped. As a result, the multiplier's complexity in terms of the number of logic cells and the lengths of critical routes is significantly decreased.

Keywords:SDLC, Approximate multiplier Design,ripple carry adders, full adders, half adders, Kogge stone adders,carry-save adders

I. Introduction

Applications including computer vision, speech recognition, and medical imaging have all shown a strong interest in computing digital signal processing and machine learning algorithms on a battery-operated device. One of the primary design criteria for these power-constrained devices is energy minimization. Additionally, it is tough to increase energy/power efficiency without sacrificing performance (speed). Multiplication is one of the most often used processes in these applications. Therefore, increasing multipliers' energy efficiency is essential for increasing the system's total energy efficiency. It's interesting that a lot of applications for machine learning and signal processing don't necessarily call for an accurate calculation.

Either the exact result is not necessary for the system to operate well, or the imperfect results frequently yield outcomes that are suitable for human perception. As a result, these systems can withstand a certain degree of inaccuracy. By applying approximation computing, this error-resilient trait is frequently used to increase the multipliers' energy efficiency. The fundamental idea behind approximation computing is to switch out traditional, energy-guzzling, complicated data processing blocks for simpler ones with fewer logic gates. As a result, effective chip area and energy usage are decreased, but the processed data now has more imprecision.

According to research, most contemporary applications in the fields of digital signal processing, computer vision, robotics, multi-media, and data analytics can tolerate some degree of such imprecision. For the current and upcoming generations of application-specific systems, this presents an opportunity to develop energy-efficient systems. For two main reasons, multipliers are essential arithmetic units in many of these applications. First of all, they are one of the most energy-hungry data processing units of contemporary microprocessors due to their complicated logic design.

Second, to compute results, compute-intensive apps frequently need a lot of multiplication operations. Since enhancements made to a multiplier's power or speed are anticipated to have a significant impact on trade-offs between system power and performance overall, these factors have drawn particular focus in research on approximate multiplier design. Accuracy is taken into account while using approximate computing as a design parameter for trading quickly and efficiently. In the literature, a number of approximation techniques to increase multipliers' energy efficiency have been documented. We have investigated the approximate array, Wallace-Tree, and Dadda-Tree multipliers. A 4 by 4 approximate Wallace tree multiplier, inexact Dadda multipliers based on approximative 4:2 compressors, and truncated designs of Array multipliers are a few of the approaches.

These approximate multipliers offered faster speeds and smaller powers, but they had significant approximation flaws. There has also been research into approximate multipliers based on partial-product accumulation. Recently, radix-based signed approximation multipliers with low energy consumption were reported. Studies have also been done on approximations based on truncating the input operands to lessen the overall multiplication. Chang Y et al [1] presented a dynamic segment technique (DSM) that was put forth to split input numbers into two or three potential m-bit segments.

A dynamic range unbiased multiplier (DRUM) was suggested by Chiou et al[2] where the least significant bits of the truncated values were set to one and m-bit segments were chosen depending on the leading one position of the numbers. An approximation multiplier based on truncation with low energy (LETAM)

II. Literature Review

Some of the most current research studies on multiplier designs were discussed in this area. Ullah et al[3] provide approximations for multipliers for hardware accelerators based on FPGA.

For FPGAs with soft multiplier IP cores to work well, improved designs are needed. Utilizing the architectural features of FPGAs, such as lookup table structures (LUT) and fast carry chains, this least latency, precise and, approximate soft-core multiplier tailored for generic areas reduces resource and overall delay of the critical route use of multipliers. Similar to Xilinx's Logi CORE IP multiplier, the exact unsigned and signed architecture reduces LUT use by up to 25% and 53%, respectively, for different multiplier sizes. Additionally, it is possible to reduce the unsigned approximation multiplier architectures under critical path latency by up to 51%.

In this, the multiplier was employed to cut down on the number of times the convolution operation was multiplied. This approach increases the number of adds, but the overall cost was reduced due to the multiplier's higher relative overhead when compared to the adder. They comprehend the 1D CONV engine as a four-stage pipeline. The use of approximation circuit libraries in the development of deep neural network hardware accelerators has been presented by Mrazek et al[4]. QoS for hardware-accelerated deep learning (DNN) applications that are fault-tolerant. Examples include the hundreds of approximate 8-bit adders and multipliers in EvoApprox8b. These circuits produce Pareto fronts for a variety of error metrics, power usage, and other circuit parameters. The enormous collection of estimated multipliers can be used to carry out ResNet's resilience study.

Addnet: Deep neural networks with multipliers optimized for FPGAs was presented by Faraone et al[5]. The optimum solution to conserve silicon space utilizing low-precision arithmetic in this manner is suggested by the tested reconfigurable constant coefficient multipliers (RCCM). By constraining the choice of coefficients, RCCMs only use adders, subtractors, bit shifters, and multiplexers to multiply the input values (MUX). The likely coefficient representations of RCCMs are mapped using unique training techniques to reduce the loss of quantification information. to demonstrate the benefits of the approaches using the ResNet-18, ResNet-50, and AlexNet networks. This low-resource RCCM beats 6-bit fixed point precision because it can achieve accuracy that is at least as good as 8-bit uniformly quantized design throughout all of its executions. The design of the logarithmic multiplier with radix-4 booth encoding was reported by Pilipovi et al[6].

A binarized neural network (BNN) for FPGAs was introduced by Liang and Yin [7]. It significantly reduces hardware usage while maintaining acceptable precision. Through data quantization and optimal storage on chip, one technique of resource-aware model analysis (RAMA) is to eliminate restricted access to bit-level XNOR multipliers and variable operations, as well as the bottleneck of parameter access. the Multilayer Perception (MLP), Cifar-10 ConvNet, and AlexNet FP-BNN accelerator designs on Stratix-V FPGA hardware. In 2019, Shanmuganathan and Brindhadevi[8] gave a summary analysis of many low effective power multiplier types. The author examines many multipliers, including Baugh's wooley multipliers, Wallace's tree multipliers, and Array multipliers. Physical verification of all of its surrogate blocks was carried out to evaluate their suitability, capacity, and low power requirements.

III. PROPOSED APPROXIMATE MULTIPLIER DESIGN

Without loss of generality, let us consider two N bit binary inputs for an (N × N) multiplier, the multiplicand (A = aN−12N−1 +⋯+a0) and the multiplier (B = bN−12N−1 + ⋯+b0). The product P can be expressed as:

$$P = A \cdot B = p_{2N-1}2^{2N-1} + \cdots + p_0 = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} a_i b_j 2^{i+j}$$

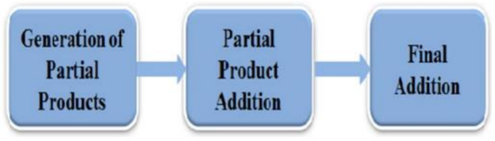
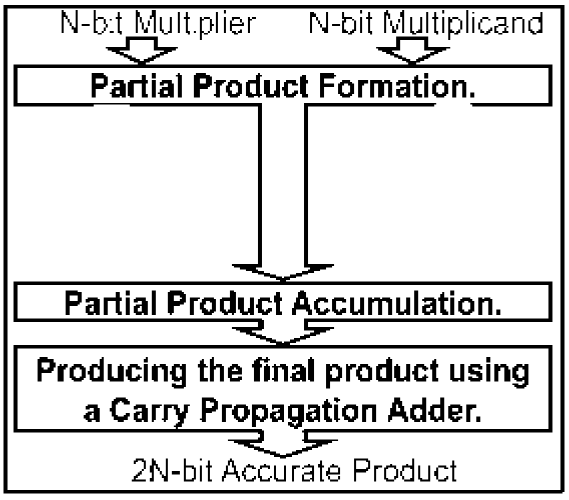
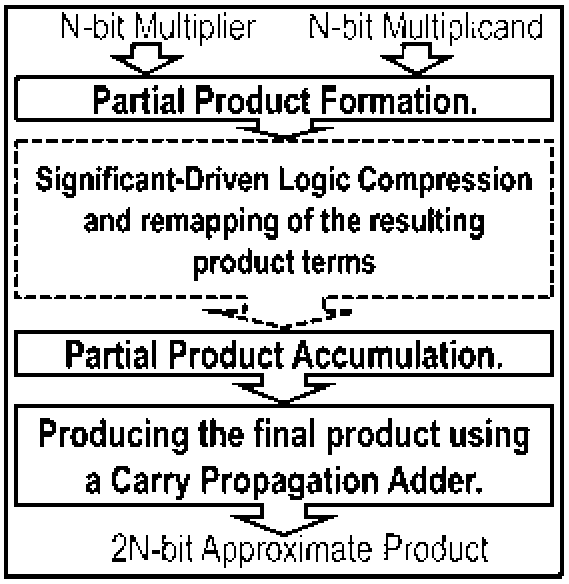


Figure.1 Block diagram of multiplier architecture

The most recent 2N-bit product is then created by synthesizing the matrix in a column using a variety of aggregation techniques, such as carrying-save range, Wallace, and dada-tree, as well as a carrying-propagated adder. The greatest tree height gathered determines the multiplier building's efficiency, software complexity, and energy requirements. In order to lower the height of the key pillar in the tree of concentration, the suggested technique suggested reducing the number of vertical components of the item in the PPM. Taking the two significant actions shown in Figure 2 (b). In first, loss compression is carried out through circuit clustering. Then, utilizing their commutative properties, the following condensed conditions are revised. These elements of the suggested design strategy are presented together with the various storage technologies.


N-bit Multiplier
N-bit Multiplicand
Partial Product Formation.
Partial Product Accumulation.
Producing the final product using a Carry Propagation Adder.
2N-bit Accurate Product

(a)


N-bit Multiplier
N-bit Multiplicand
Partial Product Formation.
Significant-Driven Logic Compression and remapping of the resulting product terms
Partial Product Accumulation.
Producing the final product using a Carry Propagation Adder.
2N-bit Approximate Product

(b)

Figure 2: Process diagram: (a) conventional math (b) In Accurate multiplication strategy

A. Significance-Driven Logic Compression (SDLC):

Figure 2(b) illustrates how this step, which can be compared to conventional multiplication, begins to create all finished items utilising N2 AND doors. Loss logic compression is used to reduce the number of pieces in the partial product matrix before moving on to the storage step. Prior to accumulation, it is intended to reduce the number of rows in the PPM to produce hardware that is not overly complex. In order to achieve loss compression, we aim to achieve the following two main values.

1. *Logic Clustering:*

The recommended multiplier separates the partial product's conditions utilizing a variety of logic cores with strong drive. Each logic array is designed to handle a set of Columns with two parts in a row of temporary products, starting with the least significant elements.
The volume logic arrays (2x L) are a set of following component product conditions with a length of columns in 2 rows and are used as an example. Every (2xL) storage set has two tasks: I can create 2L partial product components using 2L AND gates, or L pairs of vertically spaced parts, between two adjacent rows.

2. *Logic Compression:*

Each item's temporary conditions within each logic unit must be compressed every quarter using the OR door range (see Fig. 2.b). In order to get a smaller sample of the partial product matrix that has already been processed, a helpful multiplication scheme should be used. Theoretically, a double-input door is sufficient to summarise two components.
The set of items within the PPM is decreased at the fault cost by employing alternate OR compressions via logical arrays if the two provisional products conditions [aibj and an i+1bj+1] are large, that is, an error occurs when aibj+ai-1bj+1 equals O i+j 2-Bit. This error is most likely caused by (1/16), presuming that the yield values ai and bj are equal and independently distributed.

3. *Progressive Cluster Sizing:*

The PPM shortens the length of the L logic clusters because its primary objective is to create an effective power multiplier with little precision loss. The more crucial components are handled with greater precision, while the tiniest components are compressed utilizing the SDLC technique. This makes it possible to collect the most crucial item conditions using the same carry-transmission approach as a traditional multiplier. The precision of the crucial components of the finished product is thus less affected. L 2-bit (r) = N - r typically provides the duration of the 2-bit compression logic cluster used to construct the L-bit range in the simulated PPM rhythm.

IV. DEVICE AND COMPONENTS CONSIDERED

It is mentioned how this analysis is based on assumptions made at multipliers of the research publications taken into account for the survey. Logic gates and adders are used in the design of multipliers. Normally, adders are used across complete arithmetic circuits. Because products are added in parts using multiplier adders. A variety of adders, including carry adders, ripple carry adders, full adders, half adders, Kogge stone adders, and carry-save adders, are used to create multipliers. From the 100 research papers analysed, half adders and full adders are primarily used for multipliers among these adders.

V. RESULTS

The simulation result of the SDLC approach of Eight different sizes of logic clusters used to compress partial products in (16x16) parallel multiplier architecture is shown in figure 3. The various device parameters obtained from the simulation are summarized in Table.1.

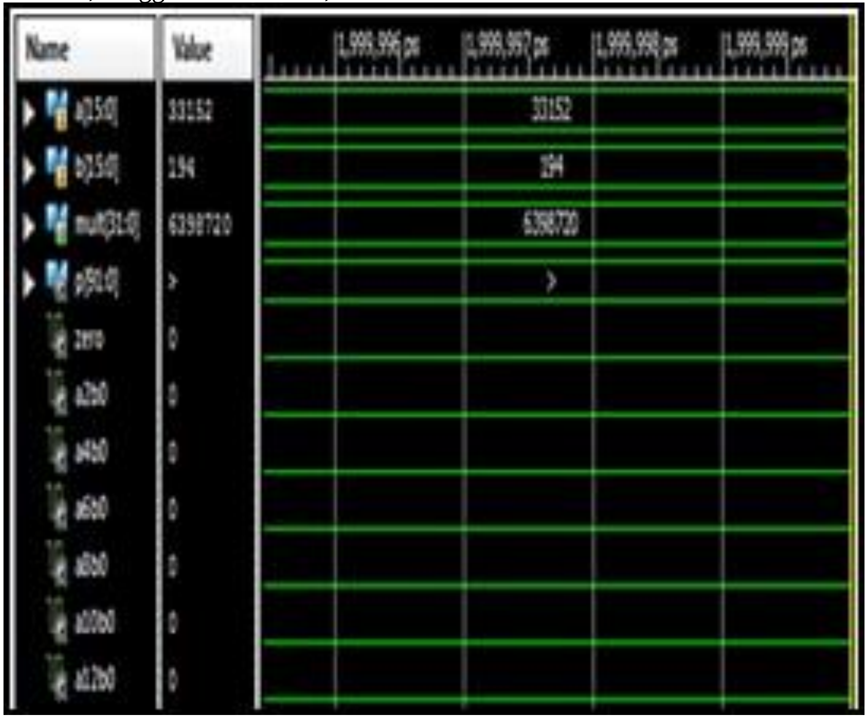


Figure.3 Simulation Result of SDLC

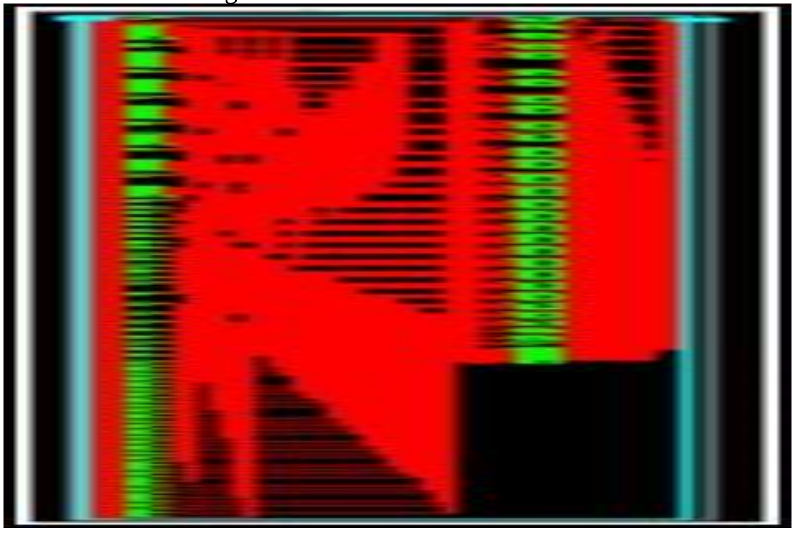


Figure.4 RTL SCHEMATIC

Table.1 DESIGN SUMMARY

Logic utilization	Used	Available	Utilization
Number of slices containing unrelated logic	0	91	0%
Number of occupied slices	91	960	9%
Number of slices containing only related logic	91	91	100%
Number of 4 input LUTS	179	1920	9%
Total number of 4 input LUTs	179	1920	9%
Number of bonded IOBs	64	66	96%

VI. CONCLUSION

A new estimated multiplier is designed using the significance-driven logic cluster technique (SDLC). This layout technique reduces the number of transient item circumstances by using an algorithmic and programmable compression of losses cantered on the part. This is then recreated and gathered utilizing various parallel computing methods. The 16x16 multiplier with SDLC offers products that are rather close to being exact for the majority of outputs. According to Xilinx experimental findings, multipliers with a fixed logic cluster size can achieve a delay of 3.590ns and an area of 86 slices as opposed to 4.221ns and 91 slices for multipliers with a variable logic cluster size. The results obtained following synthesis revealed a substantial decrease in run-time and silicon region. We believe that today's low-power computer approach can be utilized to achieve numerous benefits with little performance error.

REFERENCES

[1]Chang Y, Cheng Y, Liao S, Hsiao C ‘A low power radix-4 booth multiplier with pre-encoded mechanism’, IEEE Access 8(2020)114842–114853.
[2]Chiou C, Lee C, Lin J, ‘ Low-latency digit-serial dual basis multiplier for lightweight cryptosystems’. IET Inf Secur 11(2017)301–311.
[3]Ullah S, Rehman S, Shafque M, Kumar A , ‘High-performance accurate and approximate multipliers for FPGA-based hardware accelerators’, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems(2021)
[4]Mrazek V, Sekanina L, Vasicek Z ‘Using libraries of approximate circuits in design of hardware accelerators of deep neural networks,’ 2nd IEEE international conference on artificial intelligence circuits and systems (AICAS). (2020) 243–247
[5]Faraone J, Kumm M, Hardieck M, Zipf P, Liu X, Boland D, Leong PH ‘ Addnet: deep neural networks using FPGA-optimized multipliers’, IEEE Trans Very Large Scale Integr (VLSI) Syst 28(2019)115–28
[6]Pilipovic R, Bulic P, ‘ On the design of logarithmic multiplier using radix-4 booth encoding’, IEEE Access 8(2020)64578–64590.
[7]Liang S, Yin S, Liu L, Luk W, Wei S, ‘FP-BNN: binarized neural network on FPGA’, Neurocomputing 275(2018)1072–86
[8]Shanmuganathan R, Brindhadevi K ‘ Comparative analysis of various types of multipliers for effective low power’, Microelectron. Eng. 214(2019)28–37.