International Journal of Engineering Innovations in Advanced Technology
ISSN: 2582-1431 (Online), Volume-6 Issue-3, September 2024

ROBA Multiplier: A rounding based approximate multiplier for high-speed yet energy efficient digital signal processing

Kokkond Srinivas[1] , Dr S Kishore Reddy[2] , D Suryaprakash[3] and Davu Manvitha[4]

[1]M.Tech Student, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.

[2]Associate Professor, HOD, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., India.

[3]Assistant Professor, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.

[4]Assistant Professor, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.

Abstract **- The purpose of this study is to present a low-power approximation multiplier that is not only rapidly effective but also efficient. Rounding the operands to the closest two-exponential number is achieved by the use of this approach. Due to the reduction in the computationally expensive section of the multiplication, this results in an improvement in speed and energy usage. However, this comes at the expense of a little fall in accuracy. It is possible to handle signed as well as unsigned multiplications by using the technique that has been presented. There are three different hardware implementations of the approximative multiplier that we supply. These implementations provide signed and unsigned operations, respectively. The performance of the suggested multiplier is assessed using certain design criteria, and it is compared to the performance of comparable approximative and accurate multipliers. Furthermore, we investigated the performance of the suggested approximation multiplier in two different image processing applications, namely sharpening and smoothing operations.**

Keywords **- ROBA Multiplier,Rounding, Accuracy, Approximation, Low-power.**

I. INRODUCTION

Multipliers are one of the most basic building blocks of computer mathematics and are often used by digital signal processors. Computer graphics, scientific computations, picture processing, and many other computing applications are putting a strain on high speed multipliers. Since the speed of the multiplier determines how rapidly processors run, modern designers focus more on finding a balance between power consumption and speed. The structural components of the multiplier are an addition stage, a partial product reduction stage, and a partial product production stage. In multiplication, the partial product reduction phase is where the majority of the area, power, and latency happen. Because this step helps to decrease the critical path and, by extension, the partial products—both of which are crucial to the circuit's performance—compressors often employ it to gather partial products. do this, 3-2, 4-2, and 5-2 compressor designs are used. An other term for a 3-2 compressor circuit is the full adder cell. These compressors are integral to larger systems, thus any effort to improve their design will have a significant impact on the system's efficiency. Compressors are mostly constructed internally using multiplexers and XOR-XNOR gates. Among the several circuits that make use of these fundamental components are mathematical circuits, multipliers, compressors, parity checks, and XOR-XNOR circuits. Improving the design of these XOR-XNOR gates could boost the performance of the multiplier circuit. Here, we show how a 4-by-2 compressor makes use of a new XOR-XNOR module and detail its design. Using the proposed circuit for partial product accumulation reduces power consumption and the number of transistors.

Multiplication and addition are common tools in computer mathematics; researchers have focused on full-adder cells for approximation computing with addition. Liang et al. has assessed approximation and probabilistic adders and proposed several new metrics to measure their value in comparison to unified figures of merit for evaluating the design of inexact computing applications. For every input to a circuit, there is a predetermined mathematical distance called the error distance (ED) between the correct and incorrect outputs. The proposed mean error distance (MED) and normalised error distance (NED) provide for the normalisation of multiple-bit adders and the averaging effect of several inputs. You can use the NED to measure a design's reliability since it stays quite consistent no matter how big an implementation is. The accuracy vs. power tradeoff has also been quantitatively evaluated.

1.1 VLSI DESIGN:

This is referred to by its acronym, which is Very Large Scale Integration. Very Large Scale Integration is the process of creating integrated circuits by integrating thousands of transistor circuits onto a single chip. This process is accomplished via the use of integrated circuits.

There is a continuing trend that has substantial implications for the design of systems and for Very Large Scale Integration, and these implications are ongoing. One of the most important applications of information services refers to the ever-increasing need for very high processing power and bandwidth.

An further key use case is the provision of personalised information services, as opposed to mass-produced ones such as broadcasting. The decade of the 1970s saw the beginning of integration on a massive scale. The microprocessor is an example of a device that enables Very Large Scale Integration when used properly. The goal of this field of research is to reduce the amount of space required to accommodate more logic devices.

1.1.1 CLASSIFICATION OF VLSI DESIGNS

Within the realm of contemporary Very Large Scale Integration, there are three major types that may be found. Is that what makes them

1.1.1.1 Analog:

Amplifiers, phase-locked loops, analogue and digital converters, filters, and sensors are all examples of miniature transistor circuits that might be found in this category. Additional examples include filters and sensors.

1.1.1.2 ASIC:

Integrated circuits that are tailored to a particular purpose, often known as ASICs. Application-specific integrated circuits, often known as ASICs, are intended to perform a particular function, in contrast to general-purpose integrated circuits (ICs). Two uses of application-specific integrated circuits (ASICs) include digital voice recorders and high-efficiency Bitcoin miners. They are both examples of applications. Due to the progressive lowering of feature sizes and the gradual development of design tools, the maximum number of gates that can be included in an ASIC has climbed from five thousand to more than one hundred million. This is all due to the fact that the number of gates that may be included have increased.

1.1.1.3 SOC:

System on Chip is an acronym that describes this concept. Mixed-signal circuits that are very complex are what Systems on Chips are all about. The term "mixed signal circuit" refers to an integrated circuit that has both digital and analogue circuitry. Systems on a chip (SOC) may be represented by a variety of chips, including wireless radio chips and N/W processor chips.

1.1.1.4 Gate Array Design:

A gate array (GA) is designed in a manner that is similar to that of FPGAs. The construction of FPGA chips is accomplished by the use of user programming, in contrast to the usage of metal mask design for gate array design.

The realisation of gate array design necessitates the implementation of a manufacturing process that consists of two stages: From the beginning of the gate array design process, standard masks serve as the foundation. The implementation of complex logic functions is made possible by the use of a metal mask design that magnifies a portion of the internal array.After deleting the routing channels, it is feasible to cover the whole chip with uncommitted PMOS and NMOS transistors, similar to how Sea-of-Gates (SOG) devices are constructed. When a metal mask is applied, it is possible to modify nearby transistors in a manner that is analogous to the gate array scenario. In order for inter-cell routing to function properly, it will be necessary to reduce the number of uncommitted transistors. The use of these strategies leads to an increase in density as well as an increase in the flexibility of connection.

1.2 VLSI Design Flow:

The formal definition of a very large scale integrated circuit (VLSI) chip is the first step in the VLSI design cycle, which then extends through a variety of stages before concluding in the fabrication of a packaged chip. A typical design cycle is shown in the flow diagram that can be seen in Figure 1.

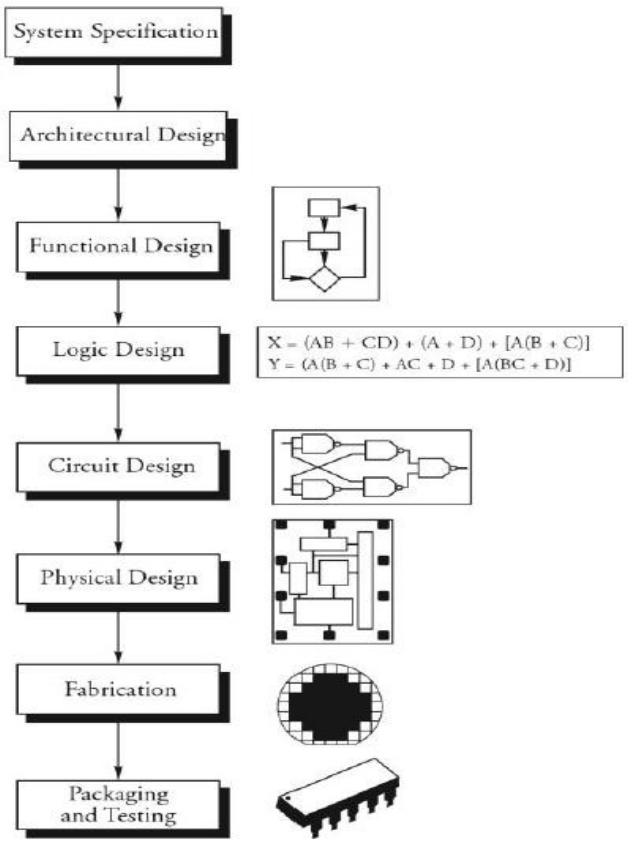


Figure 1: The Process of Very Large-Scale Integration

System Specification: The specifications of the system serve as the first step in any design process. Providing an overview of the system is the purpose of the system definition. Throughout the whole of this process, it is necessary to take into consideration the many components, including performance, functionality, and physical dimensions. The end result is a collection of requirements for the capabilities,

performance, and dimensions of the very large scale integrated circuit (VIC) system.

Architectural Design: The essential architecture of the system is built during this stage of the operation. The quantity of ALUs (Floating Point Units), the structure and number of pipelines, the size of caches, and whether or not the computer will be a RISC or CISC are some of the decisions that may be made. The culmination of architectural design is accomplished via the creation of Micro-Architectural Specifications (MAS).

Behavioral or Functional Design: At this point, the key functional components of the system are distinguished from one another. Additionally, it determines the units' requirements link to one another. It is possible to make estimates for the area, power, and other attributes of any individual unit. The most important thing is to describe how each unit should behave in terms of inputs, outputs, and time. This should be done without disclosing the inner workings of the system. It is common for functional design to result in the creation of a timing diagram or another kind of unit relationship diagram.

Logic Design: Control flow, word lengths, register allocation, mathematical operations, and logic operations are some of the elements that are included in the functional design that we construct and assess in this section of the design for the product. "Register Transfer Level" is what the abbreviation RTL stands for, and it is what identifies this description specifically. represent RTL, a Hardware Description Language (HDL) such as Verilog or VHDL is used. For the sake of simulations and verifications, you are free to utilise this description.

Circuit Design: A representation of the circuit is constructed using the logic design as the foundation. This is what the circuit design is all about. When the power and speed constraints of the initial design are taken into consideration, the Boolean expressions are turned into a representation of a circuit. Through the use of circuit simulation, it is possible to verify the precision and timing of their respective components. accurately depict the design of a circuit, the most common form of representation is a thorough circuit diagram. This diagram illustrates the components of the circuit as well as the connections linking them together.

.**Physical Design:** The circuit model is translated into its geometric equivalent throughout this step of the process's development. The diagrammatic representation of a circuit is referred to as a "layout" in the industry. The first stage in the process of constructing the layout is to create a geometric representation of each logic component that is capable of performing the logic function that it was designed to do. Geometric patterns, which often consist of lines that are arranged in several layers, may also be used to show relationships between different components. By specifying criteria that are based on the electrical properties of the fabrication materials and the limits of the fabrication process, design rules control the details of the layout. Additionally, design rules determine the layout.

.**Fabrication:** It is possible to start the fabrication process once the design has been checked and laid out. Due to the fact that layout data is often transferred to fabrication on a tape, the occurrence of data release is referred to as "Tape Out." As a result of the layout data for each layer, a photolithography integrated circuit mask is produced. The process of fabrication is comprised of a number of stages, each of which involves the deposition and diffusion of various materials onto the wafer. Additionally, the use of a different mask is required for each step. The use of a number of masks can be required bring the manufacturing process to a successful conclusion.

Packaging, Testing and Debugging: After that, the wafer is chopped into smaller pieces in a fabrication factory, which is the last stage in the process. Subsequently, each and every chip is placed into its own packaging and put through a series of tests to ensure that it functions as planned and passes all of the design criteria.

II. LITERATURE SURVEY

The purpose of this section is to offer a brief summary of some of the previous research that has been conducted on approximation multipliers. For the purpose of suggesting an approximation of both the multiplier and the adder in, the broken-array multiplier (BAM) approach was used. Through the use of the BAM approximation strategy, a signed approximation of the conventional modified Booth multiplier was presented in. In comparison to a conventional Booth multiplier, the approximation multiplier was able to cut the amount of power used by 28–58.6% and the amount of space required by 19.7–41.8 percent for words of multiple lengths. In comparison to an exact multiplier, it was shown that an approximation multiplier, which was presented by Kulkarni et al., conserved as much as 31.8% to 45.4% of the power. This multiplier consisting of a large number of 2 2 erroneous construction parts was constructed. The objective of its development was to serve as an approximation signed 32-bit multiplier in pipelined processors for the purpose of the purpose of speculation. Within a margin of error of around 14%, it outperformed a tree multiplier that was based on a complete adder by a factor of 20%.

An error-tolerant multiplier was published in, which provided the accuracies for a variety of bit widths and computed the approximation result by dividing the multiplication into two parts: one that was correct and one that was approximate. For a 12-bit multiplier, it was shown that there was a power savings of over fifty percent. Within the realm of image processing applications, the idea of using approximation multipliers has been discussed in the published literature. When compared to more precise multiplier designs, these multipliers have a lower power consumption, a lower latency, and a lower transistor count. For the purpose of developing systems that are able to tolerate errors, an accuracy-configurable multiplier architecture, also known as an ACMA, was developed. Carry-in prediction is a pre-computation logic-based method that was used by the ACMA improve its overall system performance. When compared to the accurate method, the approximation

multiplication that was recommended reduced the amount of delay by approximately half. This was accomplished by reducing the crucial path. On top of that, Bhardwaj and his colleagues presented the AWTM, which is an acronym that stands for an estimated Wallace tree multiplier. The carry-in prediction was used once again, which resulted in a reduction of the critical path. AWTM was used in a real-time benchmark photo application, and the results of this study were compared to those of an accurate Wallace tree multiplier (WTM) structure. The researchers discovered that AWTM lowered the amount of power used by forty percent and the amount of area consumed by thirty percent, all without compromising the quality of the image.

III. EXISTING DESIGN ARCHITECTURE

3.1 MULTIPLIERS

At this point in time, multipliers are an essential component in a wide variety of industries, including digital signal processing. While concurrently attaining high speed, low power consumption, and regular architecture, the design aims of a multiplier should be to decrease the size requirements as much as possible. As a result, they are perfect for a wide range of applications using VLSI. According to figure 3.1, the fundamental way of performing a multiplication is based on the iterative adder-accumulator for the formed partial products. This approach is the foundation of the procedure. One of the names that has been given to this multiplier is "serial multiplier." Unfortunately, the whole answer is not available until 'n' clock cycles have elapsed, where 'n' is the size of the operands. This approach is quite slow because of this limitation. When it comes to employing serial multipliers, space and power efficiency are of the utmost importance; yet, a little delay is quite acceptable. Through the simultaneous addition of partial products, iterative multipliers may be made to operate more quickly. By expanding the iterative multiplier, which would make it possible to achieve this goal, it is possible to build a combinational circuit that is made up of a large number of parallel adders and partial product generators. A representation of this multiplier, which is referred to as the Parallel Multiplier, may be seen in Figure 2.

25

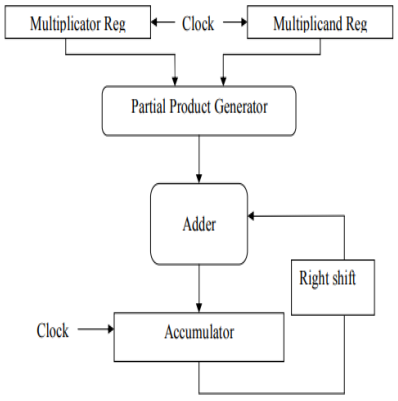


Figure 2: Repeating multiplier

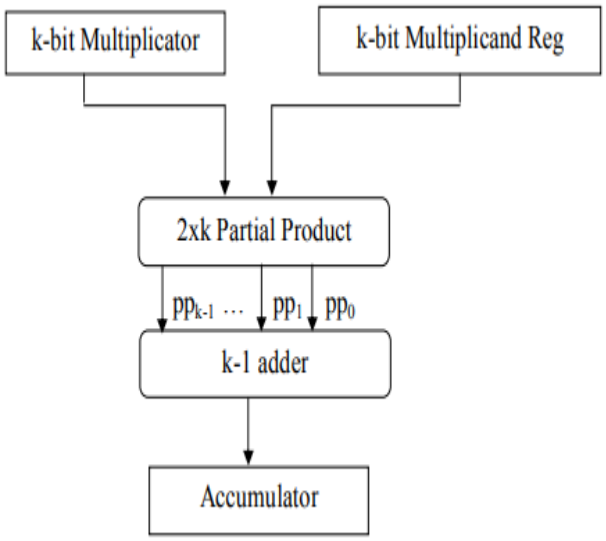


Figure 3: Parallel multiplier

The product is that which is obtained by multiplying a multiplicand by a multiplier. It is generally accepted that the process of multiplication may be broken down into two distinct stages. First things first, generate the partial product, we need to AND the multiplicand bits with the multiplier bits from the beginning. Each subsequent partial product is moved one bit position to the left in comparison to the partial product that came before it. During the second step, the partial products are accumulated by adding them up to arrive at the final result, which is referred to as the partial product accumulation. Multiplication algorithms can be broken down into two primary categories: array multipliers and tree multipliers. A full analysis of the different multipliers is presented in the parts that are to follow.

3.2 TYPES OF MULTIPLIERS

There are several types of multipliers
They are,

- Array Multiplier
- Carry save array multiplier
- Wallace Multiplier
- Dadda Multiplier
- Booth Multiplier

3.3 BOOTH MULTIPLIER

The notation of two's complement can be employed to multiply two signed binary numbers using Booth's multiplication method. In 1950, while conducting crystallographic research at Birkbeck College in Bloomsbury, London, Andrew Donald Booth developed the method.(1) One In the field of computer architecture, Booth's methodology is regarded as intriguing. The signed two's complement form of the 'N'-bit multiplier Y is analyzed using the approach devised by Booth. This form contains an implied

bit that is situated beneath the least significant bit, which is represented as y 1 = 0. The bits yi and yi−1 are considered for each bit yi. The inclusive range of i is from 0 to N − 1. When these two values are equivalent, the product accumulator P does not endure any modifications. The value of P is increased by multiplying the multiplicand by two times the value of i when the value of yi is equal to zero and the value of yi−1 is equal to one. The value of P is reduced by multiplying the multiplicand by 2i in the event that yi is equal to 1 and yi−1 is equal to 0. The signed product ultimately determines the value of P.

Despite the fact that any number system that allows for addition and subtraction may be employed, the multiplicand and product representations are not specified. Additionally, they are frequently expressed in the form of two's complement, which is quite comparable to the multiplier. This context does not specify the order of the actions. Typically, it commences its journey from the lower-left bit (LSB) to the upper-right bit (MSB) with the value of i equal to zero. Consequently, the P accumulator is incrementally moved to the right between stages, as opposed to being multiplied by 2i. This enables the manipulation of the upper N bits of P, allowing for future additions and subtractions to be performed without the low bits being in the way. The process is frequently defined as the conversion of sequences of ones in the multiplier into a high-order +1 and a low-order -1 at the extremities of the sequence. In the absence of a high-order +1, the interpretation of a string is transformed into a negative of the correct value as it traverses the MSB.

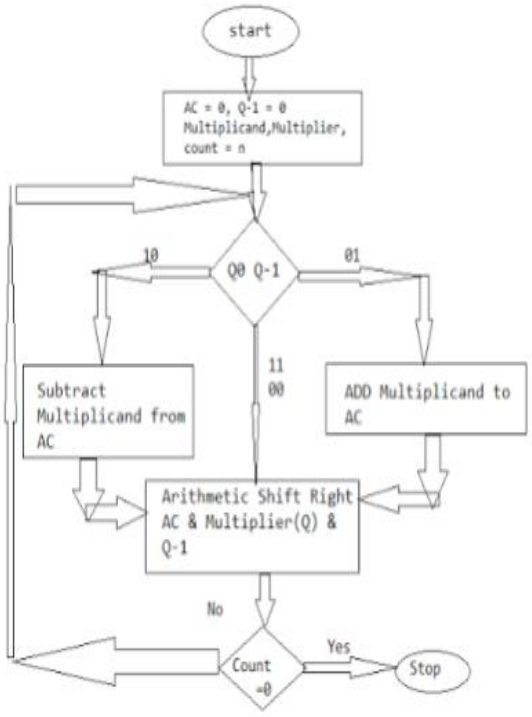


Figure 4: The booth algorithm's flowchart

Take the following example into consideration: multiplying two by four. 0010 is the signed binary form of the number 2, whereas 1100 is the signed binary form of the number -4.

• If a binary number has the fewest transitions (from 0 to 1 or 1 to 0), then it is selected as a multiplier. On the other hand, if it has the greatest changes, then it is selected as a multiplicand. One hundred is used as a multiplier in the preceding example because it only has one possible transition, which is from one to zero. On the other hand, 0010 is used as a multiplicand because it has two possible transitions, which are from zero to one and from one to zero.

In light of the fact that Multiplicand Y equals 0010 and Multiplier X equals 1100, the complement of Y is denoted by the expression'-Y' means 1110. '-Y' here is the representation of the two's complement for the Y-Multiplicand formula.

The only thing you should be doing is the arithmetic operation when the time arrives to 00 or 11. You should either add Multiplicand(Y) to AC or deduct Multiplicand(Y) from AC when the time arrives 01, respectively. After that, set AC equal to zero and X-1 equal to zero. Comparison of the LSB X0 and X-1 is the last step.

Our output is 11111000, which is equivalent to -8. This is the result we obtain.When it finds the starting digit of a one-digit block (0 1, 0 0), Booth's method adds, and when it finds the end of the block (1 0, 0), it subtracts. This is a continuation of the strategy that was taken before. It is possible to employ even a multiplier that is negative in this manner. The regular multiplication algorithm is able to do a greater number of additions and subtraction operations than Booth's approach can when the ones of multipliers are grouped together into huge blocks.

IV. PROPOSED DESIGN ARCHITECTURE

power source electrical Reduced size is one of the most important design considerations for almost all electronic systems, but it is especially important for portable electronic systems such as smartphones, tablets, and other devices. It would be exceedingly desirable to achieve this minimisation with as low of an effect as possible on performance (speed). These portable electronic devices are powered by digital signal processing (DSP) blocks, which are responsible for managing their multimedia capabilities. Mathematical logic units, which are at the core of the processing that these blocks do, are accountable for the great majority of the arithmetic operations that are performed by DSP systems. The most prevalent of these mathematics operations is multiplication. Consequently, the speed of multipliers and the power/energy economy aspects of the multipliers are the most important factors in improving CPU efficiency. In digital signal processors, the processing of still images and moving images takes up a significant portion of the available cores. Through the use of these algorithms, end goods are produced that are suitable for eating by humans. Due to the presence of this characteristic, we may make use of approximations enhance the speed and energy efficiency. When it comes to watching visual material, humans have restricted visual processing capacities, which is the root cause of this phenomenon. When it comes to some applications, such as those that deal with image and video processing, as well as other situations, the accuracy of the mathematical procedures is not an

essential need for the system to function properly. The ability to employ approximation computing gives the designer the ability to achieve a balance between a number of different parameters, including speed, accuracy, and power/energy usage, amongst others. The approximation may be used to the arithmetic units at a number of different levels of design abstraction, including as algorithms, software, architecture, logic, and circuits. It is possible to produce the approximation by the use of a variety of tactics, such as altering the Boolean function of a circuit or allowing certain timing violations (such as voltage over scaling or over clocking). There are several approaches for approximating arithmetic building blocks, such as adders and multipliers, at a variety of design levels, which are included in the class of methods for approximating functions. This paper proposes an approximation multiplier that is low-power/energy, high-speed, and is well-suited for applications that need error-tolerant digital signal processors. Assuming that the input numbers are rounded, the recommended approximation multiplier is constructed, and it is also efficient in terms of area utilisation. This is accomplished by applying some algorithmic tweaks to the conventional multiplication operation. This particular kind of multiplier is referred to as a rounding-based approximation, or RoBA for short.

We provide three optimal designs for the multiplication approach that was proposed, and it is applicable to both signed and unsigned variations of the multiplication operation. We are able to assess the effectiveness of the structures by comparing the areas, delays, power and energy consumption, energy-delay products (EDPs), and exact multipliers. The following is a summary of the contributions that the article makes regarding: (1) the presentation of a new approach for RoBA multiplication that is an alternative to the conventional way of multiplication; 2) the presentation of three hardware designs for the proposed approximate method of multiplication for both signed and unsigned operations

4.1 The Roba Multiplier Multiplication Algorithm

One of the principles that underpins the approximation multiplier that has been proposed is the idea that numbers that are raised to the power of n (2n) are particularly simple to manipulate. Let's assume that Ar and Br are the rounded input numbers for A and B, respectively, before we go on to discussing the operation of the approximation multiplier. A rephrasing of the sentence that represents the process of multiplying A by B is conceivable.

$$A \times B = (A_r - A) \times (B_r - B) + A_r \times B + B_r \times A - A_r \times B_r .$$

do the multiplications of Ar × Br, Ar × B, and Br × A, you may use the shift operation. This is the most important factor to take into consideration. However, the hardware implementation of the equation (Ar − A) × (Br − B) is highly complex and requires a lot of effort. As a result of the fact that the conclusion is determined by the differences between the accurate and rounded values, this term often has a little effect on the final output. It is for this reason that we propose eliminating this step simplify the process of multiplication. Therefore, carry out the operation of multiplication, this term is often used.

$$A \times B \cong A_r \times B + B_r \times A - A_r \times B_r$$

The objective of this strategy is to get the values of A and B that are the second-nth closest to one another. When A or B is an arbitrary positive integer whose value is larger than one, two nearby numbers, 2p and 2p−1, with equal absolute differences, are 2n and 3 × 2p−2. Because selecting the larger number (with the exception of p = 2) leads in a smaller hardware implementation for obtaining the nearest rounded value, it is taken into account in this study. This is despite the fact that neither of the two values has any effect on the accuracy of the proposed multiplier.

The concept originates from the fact that when rounding up or down, values represented as 3 × 2p−2 are considered to be reckless. This reduces the complexity of the operation and enables shorter logic statements to be used when rounding up. Three is the only variable that does not adhere to this criteria, and the approximation multiplier that is offered uses two as the number that is closest to meeting this requirement.

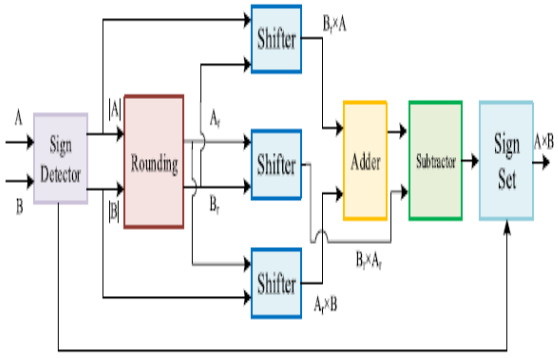
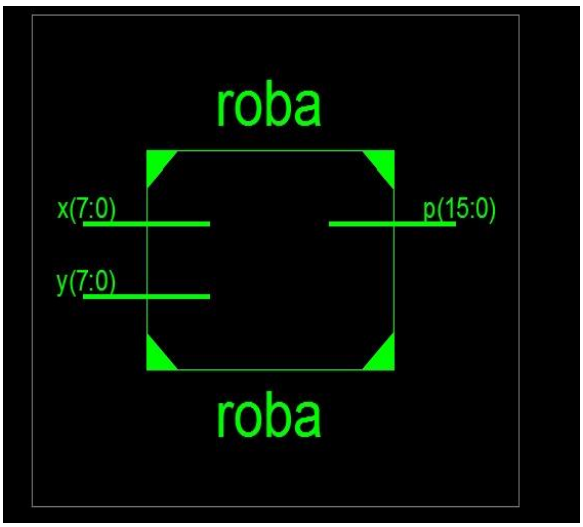


Figure 5: Schematic depicting the ROBA multiplier's hardware implementation

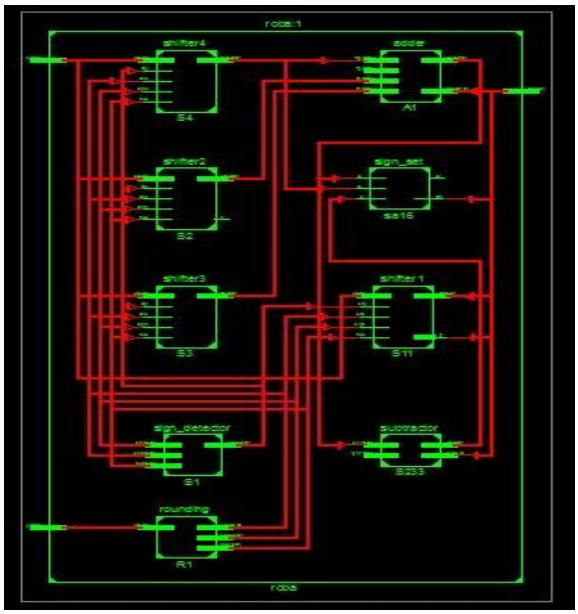
The final result of the RoBA multiplier may be higher or less than the exact result, depending on the relative magnitudes of Ar and Br in regard to A and B. It is essential to keep this fact in mind since it may have a significant impact on the outcome. On the other hand, earlier research produced an estimated result that was lower than the exact result. This stands in sharp contrast to the findings of the current study. It is possible that the estimated result will be greater than the exact result in the case that one of the operands, let's say A, is lower than its corresponding rounded value and the other, let's say B, is higher than its corresponding rounded value. In this scenario, the actual result will be less than the estimated result. This is something that must be kept in mind. This is due to the fact that in this particular circumstance, the result of multiplying (Ar – A) by (Br – B) will be a negative value.

V. RESULTS AND CONCLUSION

5.1 Proposed results



Rtl schematic



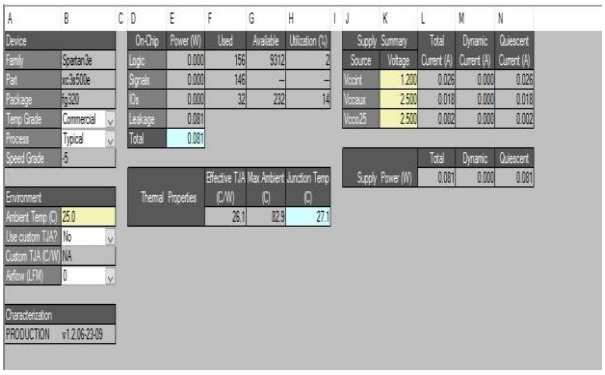
Internal block diagram

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	156	9,312	1%	
Number of occupied Slices	88	4,656	1%	
Number of Slices containing only related logic	88	88	100%	
Number of Slices containing unrelated logic	0	88	0%	
Total Number of 4 input LUTs	156	9,312	1%	
Number of bonded IOBs	32	232	13%	
Average Fanout of Non-Clock Nets	4.16			

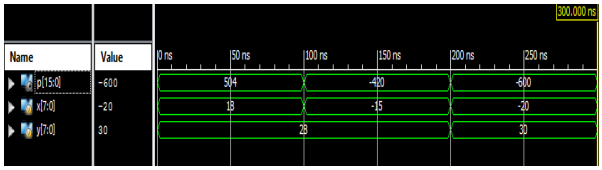
Area

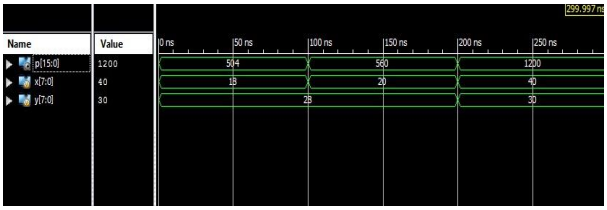
```
---------------------------------------
Total                     21.412ns (13.455ns logic, 7.957ns route)
                                   (62.8% logic, 37.2% route)

=========================================================================


Total REAL time to Xst completion: 19.00 secs
Total CPU time to Xst completion: 19.02 secs
```

Delay



Power





VI. CONCLUSION

Within the framework of this study, our RoBA multiplier is a rapid and energy-efficient approximation. Because the suggested multiplier relied on inputs rounded to the closest 2n, the findings were quite accurate. By skipping the multiplication step—which requires a lot of computing power—this strategy improved efficiency and energy usage. But a little mistake was the price it paid for that success. The given method was used to successfully finish the signed and unsigned multiplications. Although two hardware implementations were appropriate for unsigned operations, only one was appropriate for signed operations. The topic of discussion was this estimated multiplier. A wide range of precise and approximate multipliers were tested, each with its own set of design parameters, and their performance was evaluated against that of the suggested multipliers. In every case, the results showed that RoBA multiplier designs outperformed matching approximation multipliers.

Following this study, a fir filter using a roba multiplier was developed. The main advantage is that it saves time.

REFERENCES

[1] M. Alioto, "Ultra-low power VLSI circuit design demystified and explained: A tutorial," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59, no. 1, pp. 3–29, Jan. 2012.

[2] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[3] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[4] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in Proc. Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.

[5] F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in Proc. 17th Int. Symp. Comput. Archit. Digit. Syst. (CADS), Oct. 2013, pp. 25–30.

[6] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in Proc. 24th Int. Conf. VLSI Design, Jan. 2011, pp. 346–351.

[7] D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.

[8] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application," in Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits (EDSSC), Dec. 2010, pp. 1–4.

[9] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," IEEE Trans. Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.

[10] K. Bhardwaj and P. S. Mane, "ACMA: Accuracy-configurable multiplier architecture for error-resilient system-on-chip," in Proc. 8th Int. Workshop Reconfigurable Commun.-Centric Syst.-Chip, 2013, pp. 1–6.

[11] K. Bhardwaj, P. S. Mane, and J. Henkel, "Power- and area-efficient approximate wallace tree multiplier for error-resilient systems," in Proc. 15th Int. Symp. Quality Electron. Design (ISQED), 2014, pp. 263–269.

[12] J. N. Mitchell, "Computer multiplication and division using binary logarithms," IRE Trans. Electron. Comput., vol. EC-11, no. 4, pp. 512–517, Aug. 1962.

[13] V. Mahalingam and N. Ranganathan, "Improving accuracy in Mitchell's logarithmic multiplication using operand decomposition," IEEE Trans. Comput., vol. 55, no. 12, pp. 1523–1535, Dec. 2006.

[14] Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available: http://www.nangate.com/

[15] H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.

[16] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.