

## Design and Analysis of Low Power High Speed Shift and Add Multiplier for Error Tolerant Applications

**M. Manikanda Prabhu<sup>1\*</sup>, K. B. Ramesh<sup>2</sup>**

<sup>1</sup>UG student, Department of Electronics and Instrumentation Engineering,  
RV College of Engineering, Bengaluru, India

<sup>2</sup>Associate Professor, Department of Electronics and Instrumentation Engineering,  
RV College of Engineering, Bengaluru, India

**\*Corresponding Author**

**E-Mail Id: manikandaprabhu529@gmail.com**

### ABSTRACT

*In Adder circuit, the carry propagation from Least Significant Bit (LSB) to Most Significant Bit (MSB) is mostly responsible for the delay in the adder circuit. In modern VLSI technology errors of any kind are expected. Some errors like multimedia processing. In the circuit some glitches dissipate some dynamic power dissipation. If the glitches are removed or eliminated it improves the speed and power consumption of the circuit. For the current adder cell, the standard shift and add multiplier is an extension of error-tolerant mechanism. The objective of this paper is to design high speed adder to give an optimum result in terms of area, speed and power consumption. Various multipliers and adders are analysed and compare. Performance parameters are compared.*

**Keywords:** Least significant bit (LSB), most significant bit (MSB), adder circuit, multipliers, error tolerant technique

### INTRODUCTION

Most digital signal processing and other arithmetic-oriented applications use a multiplier as a core component of the processor. Due to energy loss and speed, multipliers are a necessary feature of many digital and non-digital systems. Power consumption should be kept to a minimum in portable analogue applications where power consumption is the most significant aspect. Lowering the number of signal changes in the system is one of the most effective techniques to reduce power dissipation.

The number of gates on a chip is always increasing in VLSI, yet the power to alter the gate is not decreasing at the same time. As a result, energy dissipation rises, more heat is generated, and various cooling devices, both costly and space-consuming, are employed to remove heat. For battery-powered gadgets like laptops and cell

phones, the most crucial element is power. Giving good enough results is more crucial than generating perfectly correct outcomes in analogue calculations. As a result, including an error tolerance mechanism into the design and testing process can yield acceptable outcomes.

To address high-speed and low-power circuits in order to integrate analogue, many additives and multipliers have been investigated. For 8-bit disconnection, repetitions based on word length reduction demonstrated a 56 percent reduction in 16-bit Wallace tree multipliers and a 31% reduction in 16-bit Booth frequency.

The few components that operate the switch are replaced in a regular serial multiplier. This approach delivers a higher position and a more accurate outcome at the cost of extended delays. Tolerance is not a concept that applies to all digital

applications. When using digital systems as control systems and not using an error-tolerant area, output signal accuracy is critical. Many Digital Signal Processing (DSP) systems, such as image processing and voice processing systems, that handle signals linked to the human senses of hearing, sight, smell, and touch, may benefit from debug-tolerant circuits.[1-3]

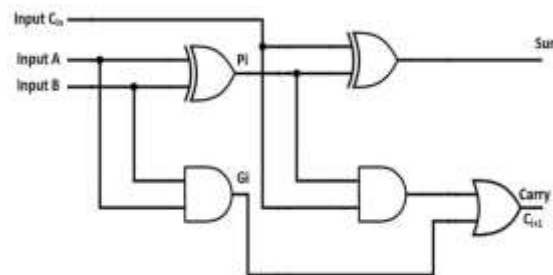
## ADDER

The essential component in any computing hardware is the adder. As a result, its performance characteristics have a direct impact on the overall system's operation. As a result, improving the performance of the adder design is a top priority. Carry save adder, carry select adder, ripple carry adder (RCA), and carry look-ahead adder (CLA) are some of the most commonly

utilised adders in digital logic processing device design. One of the most extensively utilised circuits for implementing quick arithmetic computations is the carry choose adder.[4]

## Ripple Carry Adder (RCA)

The n-bit adder made with full n-bit adders is known as the ripple carry adder because of the way the adder is calculated. Each full adder gets a  $C_{in}$ , which is  $C_{out}$  for the previous adder. This type of adder is known as the ripple carry adder because each adder loads a little "ripple" to the next complete one. The Ripple Carry Adder is shown in Figure 1 as a blockchain. In a 32-bit (ripple carry) adder, there are 32 full adders, so the critical (worst case) delay is  $31 * 2$  (forward distribution) + 3 (total) = 65 gate delay.[5]

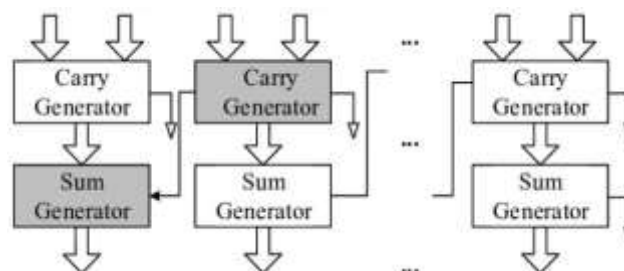


*Fig. 1: 4-bit Ripple Carry Adder.*

## Carry Look Ahead (CLA)

Carry Look Adder is a widely used design for high-speed additives on modern computers. The advantage of using a forward-looking design over the ripple carry adder is that Look-ahead incorporates the solution quickly. Carrier prices are calculated independently using a

set of sensible circuits in the forward-looking design. The ideas of producing and distributing loads are used in forward thinking. The addition of a two-digit input to both A and B is called production if it always carries, regardless of the load.  $A + B$  creates a binary addition if and only if both A and B are 1.



*Fig. 2: Carry Look Ahead Adder.*

### Error-Tolerant Adder (ETA)

The commonly used terminologies in Error Tolerant addition are as follows:

### Overall error (OE)

$$OE=|R_c-R_e|.$$

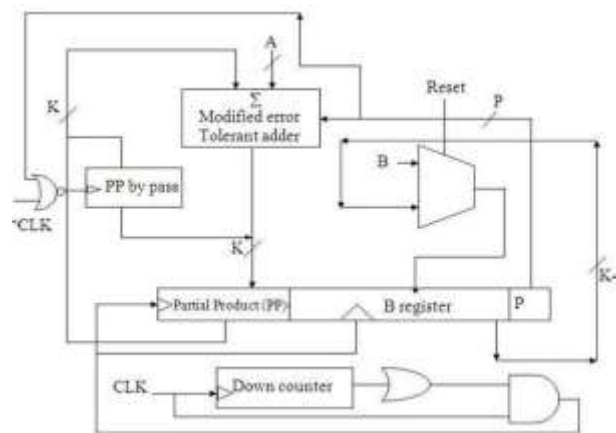
Where, Re is the result obtained by the Error Tolerant Addition technique.

Rc denotes the correct result (all the results are represented as decimal numbers).

### Accuracy (ACC)

$$ACC\%=(1-(OE/R_c))\times 100$$

In Error Tolerant Design (ETD), the accuracy of an addition process is utilized to indicate how correct the output of an adder is for a particular input. Its value ranges from 0-100%.



**Fig. 3: Error Tolerant Adder.**

## Addition Arithmetic

The delay of the normal adder circuit is caused by a transmission chain that runs on a sensitive channel from the Least Significant Bit (LSB) to the Most Significant Bit (MSB). Errors in the load distribution series also use a significant amount of dynamic energy. As a result, if load distribution can be abolished or minimised, speed and power consumption can be greatly improved.

The installation operand is split in half, with high order bits flowing in the right part and low order bits in the wrong part. The length of each episode does not need to be the same. From the design line, the add-on process proceeds in two opposing ways simultaneously.

The 8-bit input function A = 10110111 (183) and B = 10111101 (189) are divided into two 4-bit segments each with

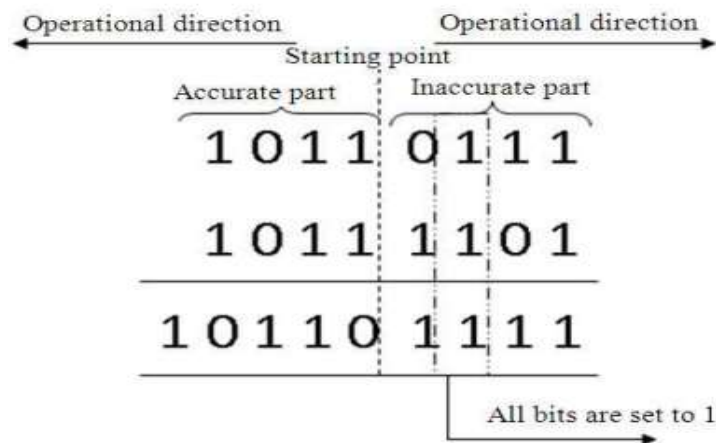
appropriate and incorrect parts in the given example. High order bits are more accurate, while lower order bits are less accurate. The correct section is made from right to left, that is, from LSB to MSB. In this case, the higher order is more important than the lower order letters. Low bit order bits are added using an error tolerance method. While the addition of low order bit bits no load will be done.

The unique method is modified to minimize total error due to termination of the bearing chain, and can be seen as follows: (1) from the right of the design line, check all areas from left to right (MSB - LSB); (2) if both input bits are "0" or different, one additional bit is added and the function proceeds to the next position; (3)

The test process is stopped if both input bits are found to be higher, i.e. 1, and all

right-bit bits (LSB) are set to "1" from this bit forward. The add-on method can be easily understood by looking at the example in Fig 3, which has the final result "10110111" (367), but should give "101110100" (372) when using standard calculations.  $OE = 372 - 367 = 5$  can be used to calculate total error. According to these two input applications, the accuracy

of the adder is  $ACC = (1 - (5/372)) \cdot 100 = 98.66$  percent. For most image processing applications, this level of accuracy is sufficient. As a result, total delay time and energy consumption are significantly reduced by removing the load method from the faulty part and making an addition to two different parts at the same time.

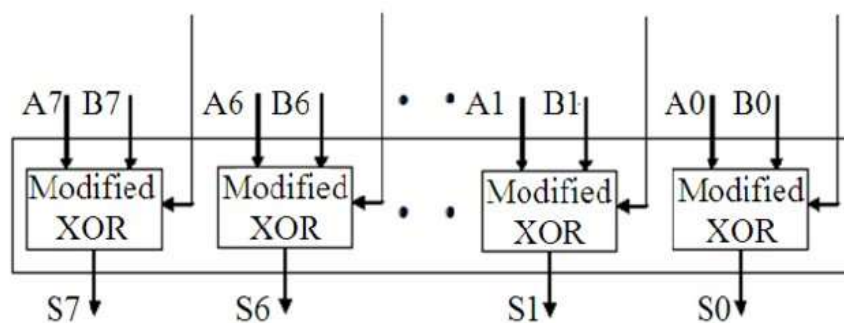


**Fig. 4:** Addition Arithmetic for Error Tolerant Design.

### Design of Accurate Part

The proposed 8-bit ETA components are featured and accurate each has four bits. The traditional cost-effective way to add ripple-carry, which is why it has been

chosen as the most accurate part of the adder circuit design. Implementation of the precision fixed part of the full adder and the modified ripple carry adder is shown in the diagrams below.

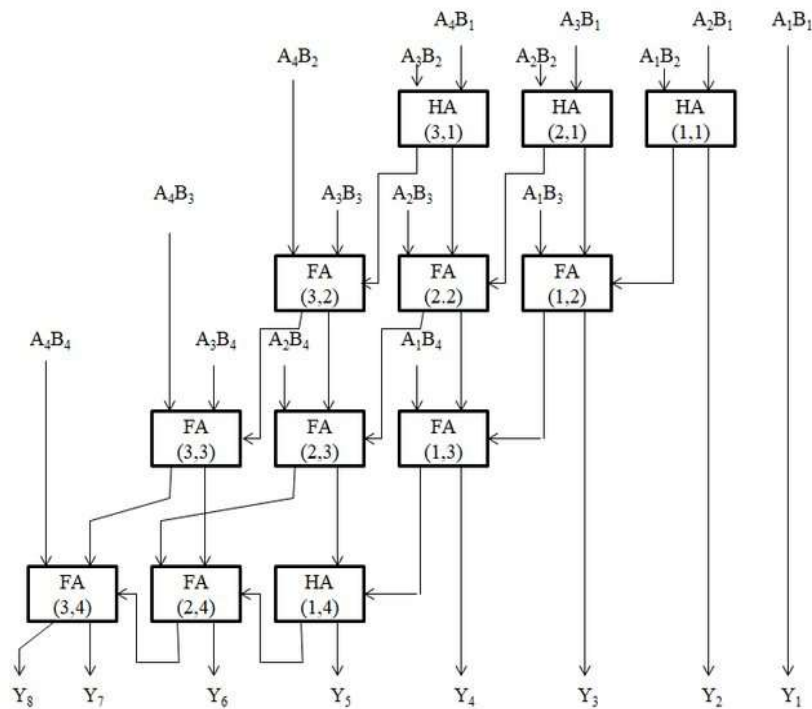


**Fig. 5:** Modified Full Adder.

### Design of Inaccurate Part

A piece of inaccurate ETA component is very important as it affects adder accuracy, speed, and power consumption. Free input block and two block control blocks from the wrong part. The free add-on block

contains four modified XOR gates, each producing a small amount of each LSB. The schematic block of carry free addition block is displayed. Fig.6 shows the implementation of the improved XOR gate system.



**Fig. 6:** Carry Free Addition Block.

## MULTIPLIER

In such systems, the multiplier is one of the most important components. These arithmetic units have a significant impact on the processor's computing speed. The multiplier's performance determines the system's performance. Different types of multipliers are available depending on the layout of the components and the system's requirements. Particular multiplier architecture is chosen based on the application.

### Booth Multiplier

The Booth multiplier uses Andrew Donald Booth's multiplication method, which multiplies two signed binary values in two's complement notation while keeping the result's sign. By lowering the number of repetition steps, it outperforms older methods of multiplication. Examining the multiplier bits, making modifications according to the algorithm, and shifting the partial product are all part of the booth algorithm. It reduces the amount of adding and subtractions required to obtain the

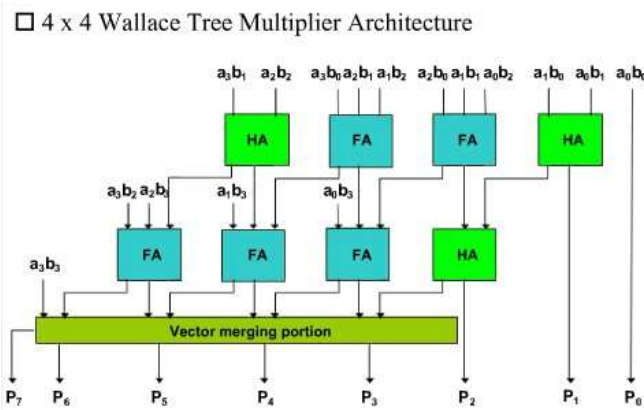
results by speeding up the multiplier operation and minimising the algorithm's chains.

### Wallace Tree Multiplier

It is an active hardware circuit designed to achieve high performance speed. Designed by Chris Wallace in 1964. Reduce the number of incomplete products and use the selected adder to add a component. Here, the total delay that occurred is accompanied by a logarithm of duration of the repetition function, which results in with rapid statistics. Wallace Multiplier is higher in terms of speed and lower power consumption than other multipliers.

The Wallace tree method of multiplication has three steps:

- The bits of one input are multiplied by the bits of the other input.
- Using half and full adders, the number of partial products is cut in half.
- The wires from the two inputs are grouped and then added together.

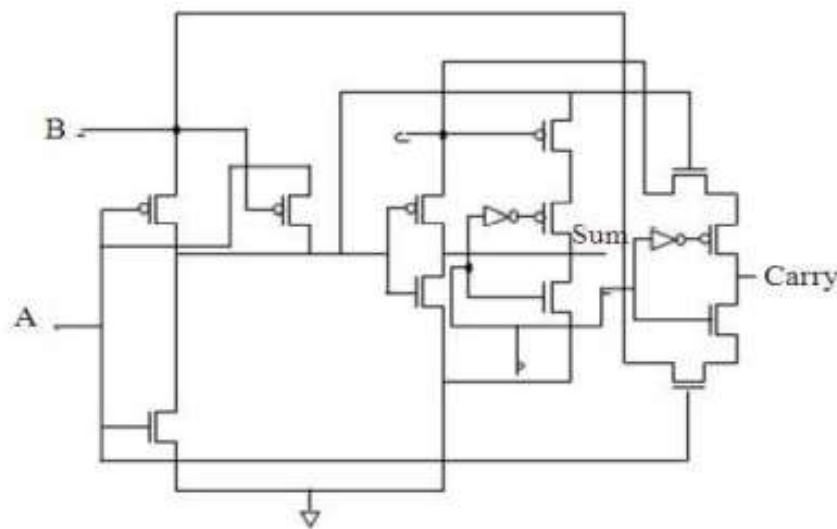


**Fig. 7:** 4-bit Wallace Tree Multiplier Block Diagram.

### Array Multiplier

It is an efficient layout of a combinational multiplier. The list of repetitive tasks is based on the one-time add-on change the algorithm. It multiplies two binary numbers using a series of half and full barriers. Adding and modifying functions are available it is done simultaneously while looking at the pieces of the multiplier following the addition of a

portion of the products. This repetition has a formal and formal structure. However, compared to other repetitions, it consumes more energy and suffers as a result major delay. Array Multiplier uses a lot of power and requires a large number of gates as a result space also increases, as a result, array duplication does not save much. Therefore, it replicates quickly but sophisticated hardware is high.



**Fig. 8:** Array Multiplier.

### Vedic Multiplier

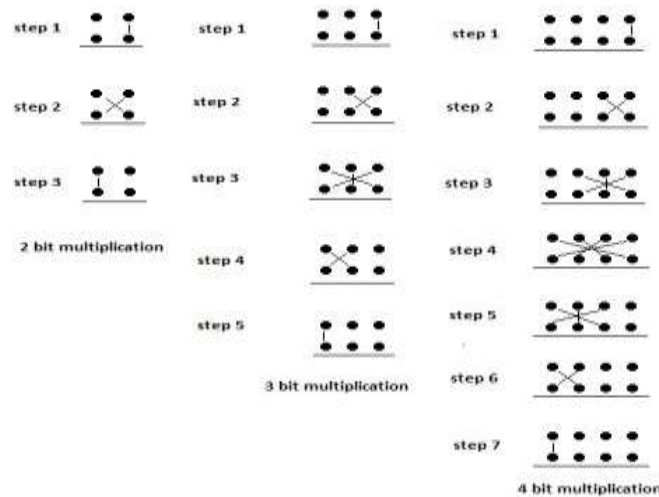
In Vedic multiplication, there are 16 sutras in total, with the Urdhva Triyakbhyam Sutra multiplier being the most efficient in terms of speed. As a result, it's also known as the UT multiplier. Both division and multiplication are covered by the Urdhva

Triyakbhyam Sutra. High-speed digital systems can be developed using the UT method in multiplier design. The development of partial products and the addition of partial products are done in parallel, resulting in fewer delays. Larger Vedic multipliers can be made by



combining numerous  $2 \times 2$  Vedic multipliers and adding the results using a ripple carry adder. Using reversible logic gates like Peres, Feynman, DKG in the design can further aid in the reduction of

power consumption. Fig. 10 shows the line diagram for multiplication of two 4-bit numbers based on Urdhva- Triyakbhyam Sutra.



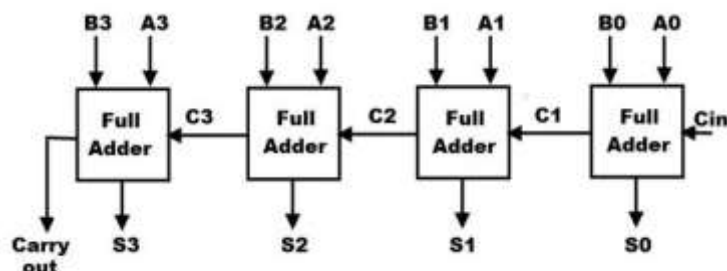
**Fig. 9:** Line Diagram based on UT Multiplier.

#### PROPOSED ET SHIFT AND ADD MULTIPLIER:

(i) Adder that is error tolerant (ii) Register for partial products (PP) (iii) Register for multiplier (B) (iv) The proposed design's major blocks are the PP bypass register and (iv) down counter.

The PP register, the B-registers with multiple bits, and the A-register with multiplicand bits are all initially set to zero. The P signal for the Error Tolerant Adder is a B (0) (least important) B-register. If  $P = 1$ , the multiplication pieces of the A register are combined with the pieces of the product register. If  $P = 0$ , the Permanent Error Adder is turned off, and

the removed pieces of the PP register are passed through the pass register from the adder. The PP and B registers are connected together using the AND signal from the lower output exit and the clock, as shown in Figure 8. The reset counter pieces will be set to the top first. The contents of the PP and B registers are rotated one bit slightly to the LSB with each decrease in the calculation values, and the exchange function is stopped when the counting pieces reach zero. As a result, the counter should be built using a bit count multiplier. Adder block switch function and multiplexer input are reduced.



**Fig. 10:** Proposed ET Shift- and Add Multiplier

## RESULT

To see how well the proposed design works, we compared it with the old version of the switch and added with the BZ-FAD properties (via Zero Feed A directly). The Electricity Delay (PDP) product, which exposes energy consumption, is another important factor to consider. Focusing on energy or delay only gives an incomplete picture because delays can often be reduced by increasing energy consumption.

The PDP can be calculated using the strength and delay data obtained. Compared to the normal multiplication of addition and subtraction, the recommended ET multiplier has 59.9% PDP and 35.3 percent PDP compared to the BZ-FAD multiplier and add-on. When the proposed output of the alternating output and recurrence ET is compared to the actual value of 1000 samples, the error fraction is determined to be 1.4 percent, indicating an accuracy of 98.6 percent. In image, audio signal, and video processing applications, this level of inaccuracy is the most tolerable.

## CONCLUSION

In this study, the concept of tolerance is applied to produce shift-and-add frequency. In order to achieve large energy savings and performance gains, the proposed frequency raises the risk of certain accuracy.

The suggested Error Tolerant Adder sacrifices greater accuracy in order to

obtain large energy savings and operational benefits. Extensive comparisons with prior additives, such as the Ripple Carry Adder, show that the proposed ETA outperforms typical additions in terms of energy efficiency.

## REFERENCES

1. Rajesh, K., & Reddy, G. U. (2019, January). FPGA Implementation of Multiplier-Accumulator Unit using Vedic multiplier and Reversible gates. In *2019 Third International Conference on Inventive Systems and Control (ICISC)* (pp. 467-471). IEEE.
2. Ozcan, E., & Erdem, S. S. (2019, July). A High Performance Full-Word Barrett Multiplier Designed for FPGAs with DSP Resources. In *2019 15th Conference on Ph. D Research in Microelectronics and Electronics (PRIME)* (pp. 73-76). IEEE.
3. Vijeyakumar, K. N., Komanduri, V. S. S. Suji, C. C. G. (2011). Science Publications. *Journal of Computer Science*, 7(12), 1839-1845.
4. Breuer, M. A., Gupta, S. K., & Mak, T. M. (2004). Defect and error tolerance in the presence of massive numbers of defects. *IEEE Design & Test of Computers*, 21(3), 216-227.
5. Sterpone, L., Sonza Reorda, M., Violante, M., Kastensmidt, F. L., & Carro, L. (2007). Evaluating different solutions to design fault tolerant systems with SRAM-based FPGAs. *Journal of Electronic Testing*, 23(1), 47-54.