International Journal of Advanced Trends in Engineering and Management
(ISSN 2583-7052)

Vol. V, No. 09, pp. 41-49, September

# Optimized Booth Multiplier Architecture for High-Speed Arithmetic Applications

Ramya R R[3]

[3]Assistant Professor, Department of Computer Science and Engineering,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,
Avadi, Tamil Nadu, India.

**Corresponding Author E-mail:**

**ABSTRACT:** The Fast and efficient multiplication has become the key for modern digital systems. High computational delay and hardware complexity are inherent problems in conventional multiplier architectures. In this work a Modified Radix-4 Booth Multiplier is presented for high-speed binary multiplication. The proposed architecture uses Modified Radix-4 Booth Encoding, a Partial Product Generator (PPG), Wallace Tree Reduction, and a Carry Look-Ahead Adder (CLA) to reduce partial products and to speed into the final addition. The partial-product generation is reduced to decrease the computational complexity and propagation delay. The simulation results confirm that the multiplication works correctly for normal, signed, zero and boundary conditions. The proposed multiplier provides a propagation delay of 6.53 ns, whereas the Array Multiplier and the Radix-2 Booth Multiplier provide 9.02 ns and 8.32 ns, respectively, demonstrating improved speed and efficient utilisation of hardware.

**Keywords:** Modified Radix-4 Booth Multiplier, PPG, WTC, CLA, High Speed VLSI.

## 1. Introduction

In contemporary digital systems, fast and energy-efficient arithmetic operations have become very important for attaining better performance in computations. Multimedia applications, image processing, digital signal processing, communications systems, and IoT are some of the fields which demand fast computation with less energy dissipation and lower hardware complexity [1-2]. From several arithmetic units, the multiplier stands out as one of the most important blocks that has been utilized in processors, embedded systems, and digital signal processing systems. Therefore, efficient architecture of multipliers has become an active research area [3].

Recently, much attention has been paid to optimal multiplier design by reducing the partial products and minimization of the corresponding accumulation delay [4]. Traditional multiplication algorithms involve multiple stages of additions, which add to the hardware cost, power consumption, and delay time. Thus, there is a need for an effective multiplication algorithm that provide the best compromise between speed, hardware implementation, and power savings. Multiplication using Booth's algorithm is one of such approaches due to its reduced number of partial products [5-6].

Various types of multiplier circuits have been designed to improve the performance parameters like speed, area, power, and delay. The sequential multipliers require lower hardware and lower power but high latency whereas the parallel multipliers such as Wallace Tree and Dadda require higher hardware and lower latency. The Array Multiplier has a regular structure but it consumes much area and delay because of partial

products generation. The Radix-2 Booth Multiplier decreases arithmetic operations and is used for signed multiplication but needs additional encoding circuits. The Dadda Multiplier reduces partial products efficiently and gives high speed but irregular connections cause problems during implementation [7-9]. To resolve these issues, the Modified Radix-4 Booth Encoder performs computations on two bits of the multiplier and hence reduces partial products.

The Modified Radix-4 Booth Multiplier reduces the number of partial products which enhances the speed and complexity reduction. The proposed 16-bit architecture incorporates Booth Encoding, Wallace Tree Reduction and CLA for fast and efficient multiplication with lower delay and hardware usage.

### 1.1 Research Gap

However, the existing multiplier architectures still suffer from a trade-off between multiplication speed, partial-product reduction and hardware complexity. Array multipliers need more partial-product operations, and traditional Radix-2 Booth multipliers offer partial-product reduction but introduce additional encoding operations. Hence there is a research gap in the development of an optimised multiplier architecture which can further reduce the partial products while minimising the propagation delay and hardware requirement. To overcome this limitation, the proposed architecture combines Modified Radix-4 Booth Encoding with a Partial Product Generator, Wallace Tree Reduction and Carry Look-Ahead Adder to achieve faster and more efficient binary multiplication.

## 2. Recent Works

| S. No | Tittle/ author name/year | Methodology used | Advantages | Disadvantages |
|---|---|---|---|---|
| 1. | Ahmed Khan *et al* (2022) [10] | The CNT based full adder is designed on 32 nm technology and simulated for Sum and Carry generation. Its power, delay and PDP are evaluated. | The design provides high speed, low power consumption, and reduced propagation delay. It is amenable for compact and energy efficient arithmetic circuits. | Challenges for CNT-based circuits include fabrication complexity, process variations and integration issues. These factors limit practical large-scale implementation. |
| 2. | Marangunic *et al* (2022) [11] | The arithmetic module on ML learns the computational pattern and optimises the arithmetic operations to achieve high-speed processing. Its performance is evaluated in terms of delay, power and hardware utilisation. | It offers fast computation, adaptive optimisation and better hardware efficiency for arithmetic operations. | Demands training data, more computational resources, and raises design complexity. |
| 3. | Smith *et al* (2022) [12] | The arithmetic module is designed to perform basic arithmetic operations efficiently using ultra-low-voltage circuit techniques. It is evaluated on basis of speed, power consumption, delay and throughput. | It offers low power consumption, high throughput and low operating voltage, making it suitable for energy efficient computing systems. | Ultra-low-voltage operation suffer from reduced noise margin, increased sensitivity to process variations and reduced reliability. |
| 4. | Padmanabhan *et al* (2022) [13] | The multiplier breaks multiplication into simpler partial-product computations for high-speed computation, partitioning binary operands into smaller groups. | It provides reduced delay, faster multiplication and efficient partial product generation. | Grouping and decomposition increase control logic and hardware complexity |

### 2.1 Objectives

- To design a 16-bit Modified Radix-4 Booth Multiplier for high-speed binary multiplication.
- To reduce the number of partial products using Modified Radix-4 Booth Encoding.
- To efficiently generate and reduce partial products with a PPG and Wallace Tree.
- To increase the final addition speed using CLA.
- To obtain an accurate 16-bit product output with improved speed and hardware efficiency.

## 3. Proposed Work Explanation

The block diagram of 16-bit Modified Radix-4 Booth Multiplier for fast and efficient binary multiplication has been shown Fig.1. It starts with two input operands A and B. These are fed into the Modified Radix-4 Booth Encoder. The groups of multiplier bits are processed by the encoder to produce optimised control signals such that the number of partial products is reduced. The encoded results are sent to the PPG that generates the corresponding partial products shifted and signed. These partial products are then fed to the Wallace Tree Reduction stage where they are compressed efficiently through parallel addition, thus reducing the overall propagation delay. The CLA finally takes care of the resulting sum and carry outputs, and does the addition quickly.

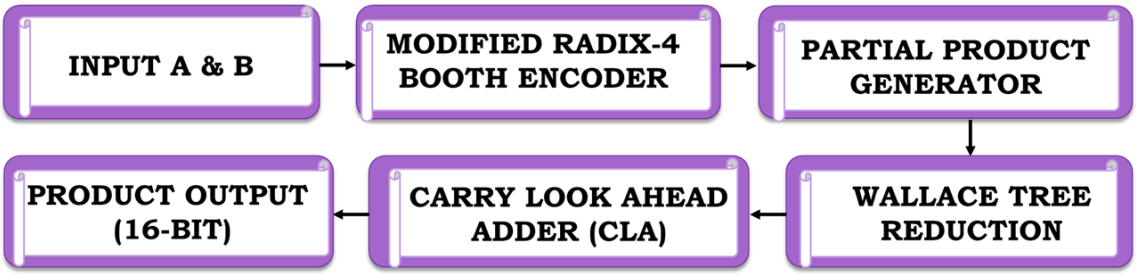



**Figure 1:** *proposed system Architecture*

The architecture produces the final 16-bit product output with enhanced speed and efficient hardware utilisation.

## 4. Proposed system modelling

### 4.1 Modified Radix-4 Booth Encoding

The Modified Radix-4 Booth Encoding is used to perform high speed and efficient binary multiplication by reducing the number of partial products. In this method, the multiplier bits are grouped into overlapping groups of three bits, with two multiplier bits being processed in each encoding step. The multiplication for an n-bit two's-complement multiplier (Y) is given by

$$X \times Y = X \times \sum_{i=0}^{[n/2]-1} M_i 2^{2i} \quad (1)$$

Where $X$ is the multiplicand and $M_i$ is the Booth encoded coefficient obtained from the three multiplier bits$(y_{2i+1}, y_{2i}, y_{2i-1})$, with$y_{-1} = 0$.

The Booth encoded coefficient is given by

$$M_i = 2y_{2i+1} + y_{2i} + y_{2i-1} \quad (2)$$

Hence, the possible values of the encoded coefficient are

$$M_i \in \{-2, -1, 0, +1, +2\} \quad (3)$$

The corresponding partial product is.

$$PP_i = (M_i X)^{2^{2i}} \quad (4)$$

So, the final result of the multiplication procedure is obtained as

$$P = \sum_{i-0}^{[n/2]-1} PP_i \quad (5)$$

Modified Radix-4 Booth technique processes two multiplier bits at a time, and reduces the number of partial-product rows by almost half over conventional binary multiplication. This reduction reduces computational complexity, propagation delay and hardware resource demand.

The proposed multiplier is implemented and verified for 8-bit input data. The operands A (7:0) and B (7:0) are processed to produce a 16-bit product, PRODUCT (15:0). The top-level RTL block diagram of the proposed optimised Booth multiplier with two 8-bit input operands and the corresponding 16-bit output is shown in Fig. 2.

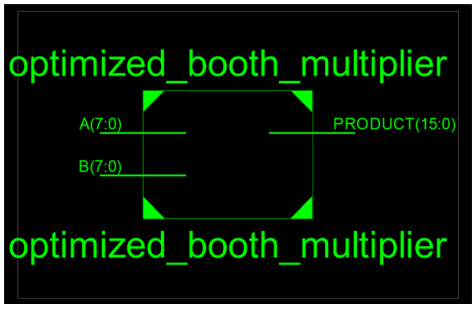



**Figure 2:** *RTL block diagram of the proposed optimized booth multiplier*

Figure 3 shows the detailed proposed multiplier RTL schematic, which illustrates the interconnection of the functional logic blocks and the signal flow involved in the optimised Booth multiplication process.

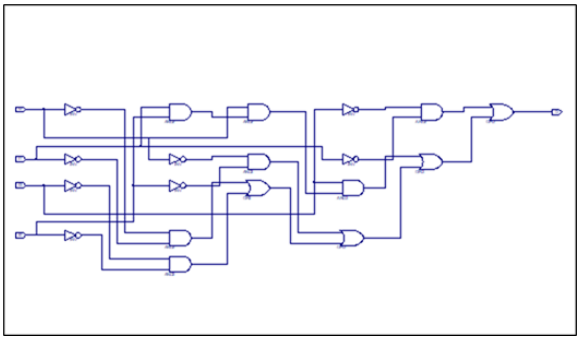


**Figure 3:** *RTL schematic diagram of the proposed optimized booth multiplier*

The Boolean logic expression of the proposed multiplier is shown in Fig. 4, which determines the logical conditions to generate the corresponding output signals.

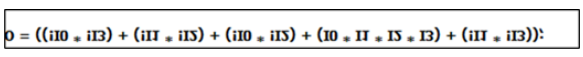
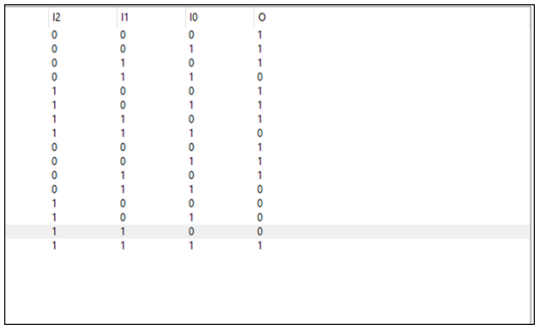


**Figure 4:** *Boolean logic expression*

The truth table for the implementation of the Boolean logic is shown in Figure 5. It shows the output O for all possible values of the input signals I2, I1 and I0 which confirms the functional behaviour of the proposed logic.



| I2 | I1 | I0 | O |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**Figure 5:** *Truth table*

Figure 6 shows the 4-variable Karnaugh map (K-map) for the proposed Boolean logic. It shows the grouping of input combinations used to simplify the Boolean expression to get an optimised logic implementation.

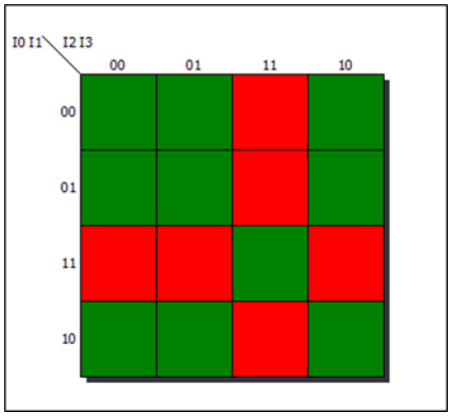



**Figure 6:** *Variable Karnaugh map*

Overall, the modified radix-4 Booth encoding reduces the number of partial products and simplifies the multiplication process. The functional operation of the proposed 8-bit × 8-bit Booth multiplier is verified using RTL implementation, Boolean logic, truth table and K-map which is suitable for high-speed digital arithmetic applications.

### 4.2 Partial product generator

The booth encoded multiplier generates partial products from the 16-bit multiplicand $X$ using the PPG. The corresponding partial product is

$$PP_i = M_i \times X \tag{6}$$

and the shift of position is written as

$$PP_I' = PP_i \times 2^{2i} \quad (7)$$

The number of Booth groups for the 16-bit multiplier is

$$N = \left[\frac{16}{2}\right] = 8 \quad (8)$$

$$P = PP_0' + PP_1' + PP_2' + PP_3' + PP_4' + PP_5' + PP_6' + PP_7' \quad (9)$$

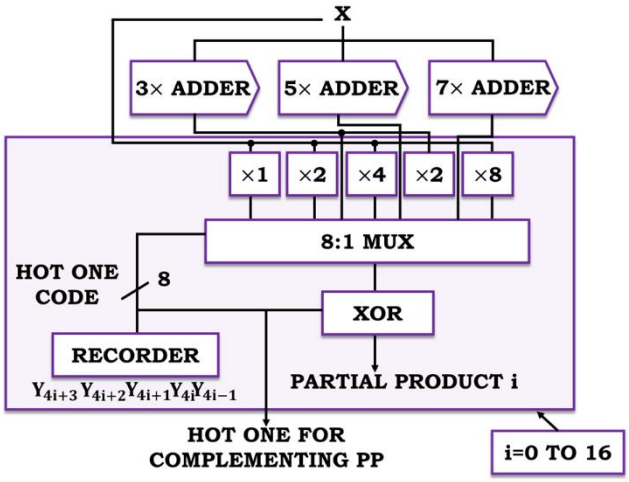



**Figure 7:** *PPG of the proposed multiplier*

Thus, instead of sixteen partial products, a maximum of eight partial products is generated. This decreases the additions and improves the hardware efficiency.

### 4.3 Wallace tree reduction

The generated partial products are reduced by the Wallace Tree Reduction stage using 3:2 and 2:2

Compressors. The partial-product matrix is compressed to two rows reducing the stages of addition and propagation delay.

The height of the partial-product matrix at the kth reduction stage is

$$H_{k+1} = 2\left[\frac{H_k}{3}\right] + (H_k mod\ 3) \quad (10)$$

Where $H_k$ is the height of the matrix at the current reduction step. The reduction goes on until

$$H_k \leq 2 \quad (11)$$

Thus, the Wallace Tree reduces the partial-product matrix efficiently with fewer compression stages and less propagation delay. Finally, the Carry Look-Ahead Adder (CLA) adds the remaining sum and carry outputs to generate the final 16-bit product, which results in high-speed multiplication with efficient hardware utilisation.

## 5. Results and Discussion

The proposed Optimised Booth Multiplier is verified for its functional correctness and hardware performance using simulation and synthesis. The results show that the proposed approach work reliably with reduced delay and efficient utilisation of resources

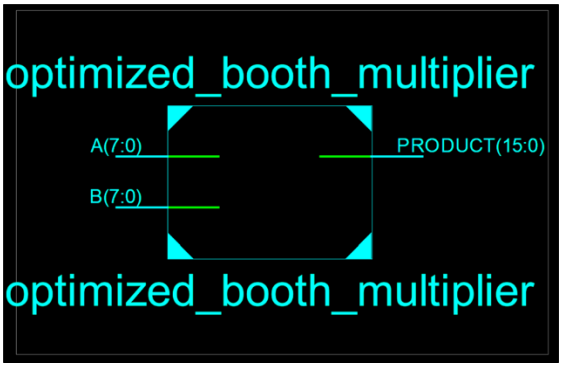



**Figure 8:** *RTL Block Diagram of the Proposed Optimized Booth Multiplier*

Figure 8 depicts the top-level RTL architecture of the proposed optimised booth multiplier. The 8-bit inputs a (7:0) and B (7:0) the multiplier multiplies these operands and generates the 16-bit product (15:0).


**Figure 9:** *Basic Multiplication Verification*

Figure 9 represents waveform for verification of the basic multiplication operation with input operands a (7:0) and B (7:0). The achieved PRODUCT (15:0) verifies the correct functional operation of the proposed multiplier.

**Figure 10:** *Consecutive Operand Verification*

Figure 10 shows waveform for a successive operand combination applied to the proposed Booth Multiplier. The corresponding PRODUCT (15:0) outputs confirm the correct multiplication operation for the input sequences applied.

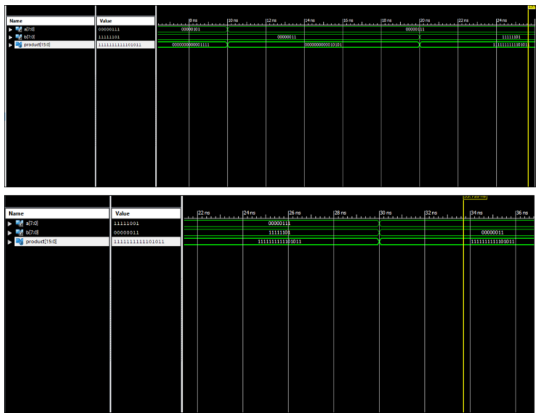


**Figure 11:** *Sign Reversal Verification*

Figure 11 indicates simulation waveforms for sign reversal conditions of multiplicand verify positive and negative operand operations. The PRODUCT (15:0) outputs confirm the correct signed multiplication behaviour of the proposed Booth multiplier.

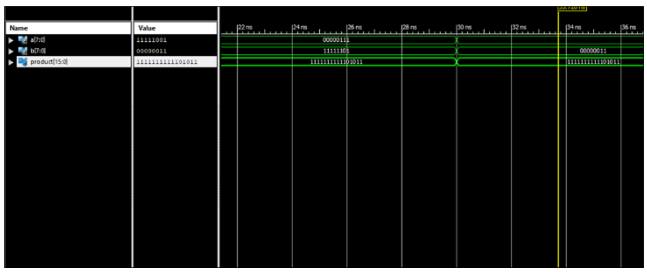


**Figure 12:** *Double Negative Verification*

Figure 12 depicts simulation waveform of the multiplication of two negative operands in the proposed Booth multiplier. The resultant PRODUCT (15:0) confirms the correct positive output for the double-negative multiplication condition.

**Figure 13:** *Higher Value Multiplication and Zero Operand Verification*

Figure 13 shows the simulation waveforms for higher value multiplication and zero operand conditions for the proposed Booth multiplier. The corresponding outputs confirm correct multiplication results including the expected zero output when one of the operands is zero.

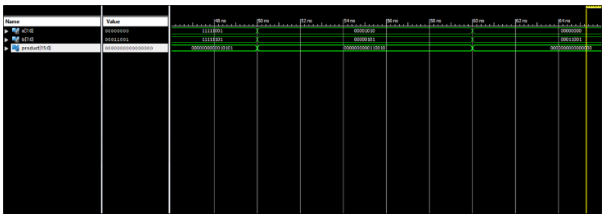


**Figure 14:** *Maximum Positive Range Verification*

Figure 14 shows the simulation waveform for multiplication using maximum positive-range operand values. The obtained output PRODUCT (15:0) shows the correctness of multiplication and justifies the arithmetic range of the proposed Booth multiplier.

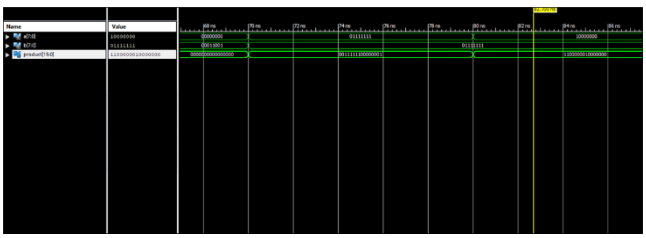


**Figure 15:** *Signed Boundary Stress Verification*

Figure 15 shows the simulation waveform for the signed boundary operand conditions applied to the proposed Booth multiplier. The outputs of that result confirm correct arithmetic operation at the signed boundary range.

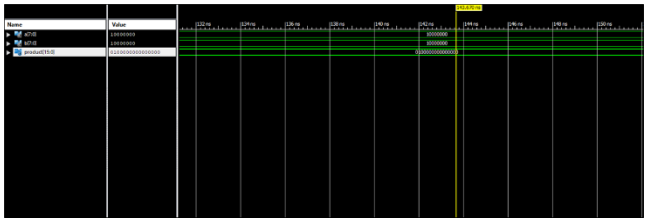


**Figure 16:** *Minimum Value Limit Verification*

Figure 16 shows the simulation waveform for minimum-value boundary operands applied to the proposed Booth multiplier. The result obtained confirms the correct multiplication at the lower arithmetic limit.

**Table 1:** *Design summary of the proposed booth multiplier*

| **Design summary** | |
|---|---|
| Top level Output File Name | Optimized_booth_ multiplier |
| Primitive and black box usage | |
| GND | 1 |
| MUXCY | 57 |
| VCC | 1 |
| XORCY | 64 |
| IO Buffers | 32 |
| IBUF | 16 |
| OBUF | 16 |

Table 1 shows the implementation resources used by the proposed optimized Booth multiplier including primitive components, black-box components and I/O buffers. The design is utilizing 57 MUXCY, 64 XORCY, 16 IBUF and 16 OBUF resources. This depicts the hardware utilization of the proposed architecture**.**

**Table 2:** *HDL synthesis report*

| **HDL synthesis report** | |
|---|---|
| **Macro statistics** | |
| #Multipliers | 4 |
| 8×3-bit multiplier | 4 |
| #Multiplexers | 20 |
| 2-bit 2 to 1 multiplexers | 8 |
| 3-bit 2 to 1 multiplexers | 12 |
| *Xors | 59 |
| 1-bit xor2 | 35 |
| 1-bit xor 3 | 23 |
| 1-bit xor 6 | 1 |

Table 2 Summarizes the synthesized hardware macros used in the proposed multiplier which consists of 4 multipliers, 20 multiplexers and 59 XOR gates. These synthesis results show the hardware requirements and internal logic utilization of the proposed architecture

**Table 3:** *Timing summary*

| **Timing Summary** | |
|---|---|
| Delay | 6.53 ns |
| source | B<3>(PAD) |
| Destination | PRODUCT< 15 > (PAD) |
| Data path | B<3> to PRODUCT< 15 > |

Table 3 Shows the timing performance of the proposed multiplier. The maximum delay is 16.205 ns from input B<3> to output PRODUCT. This result represents the propagation delay of critical data path in synthesised design.

**Table 4:** *Delay comparison of different multiplier architecture*

| **Delay** | |
|---|---|
| Method | Value |
| Array multiplier [7] | 9.02 ns |
| Radix 2 booth multiplier [8] | 8.32ns |
| Proposed | 6.53 ns |

Table 4 Shows propagation delay of Array Multiplier, Radix-2 Booth Multiplier and proposed multiplier. The delay is minimum for the proposed architecture as 6.53 ns showing the better speed than the conventional designs.

## 6. Conclusion

This paper presents a Modified Radix-4 Booth Multiplier for fast binary multiplication. The proposed architecture uses Modified Radix-4 Booth Encoding, PPG, Wallace Tree Reduction and CLA to reduce the number of partial products and propagation delay. The functional simulation confirmed the correct multiplication in the conditions of standard, signed, zero and boundary-value. The propagation delay for the proposed multiplier is 6.53ns whereas Array Multiplier has 9.02ns and Radix-2 Booth Multiplier has 8.32ns which shows the improvement in the computational speed. The synthesis results also ensure the efficient hardware utilisation. The

proposed architecture is suitable for high-speed VLSI and digital arithmetic applications. In future work, the architecture extended to higher bit multipliers with further optimisation of area, power and delay.

## References


1. Khamalesh Kumar Padmanabhan; Umadevi Seerengasamy; Abraham Sudharson Ponraj, Year: 2022, "High-Speed Grouping and Decomposition Multiplier for Binary Multiplication", Electronics, Vol: 11, No: 24, pp. 4202.
2. Tan Weihang; Antian Wang; Xinmiao Zhang; Yingjie Lao; Keshab K. Parhi, Year: 2023, "High-speed VLSI architectures for modular polynomial multiplication via fast filtering and applications to lattice-based cryptography", IEEE Transactions on Computers, Vol: 72, No: 2, pp. 2454-2466.
3. D. Bhuvana Suganthi; M. Shivaramaiah; A. Punitha; M. K. Vidhyalakshmi; S. Thaiyalnayaki, Year: 2023, "Design of 64-bit Floating-Point Arithmetic and Logical Complex Operation for High-Speed Processing", In 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), pp. 928-931.
4. Thanikodi Manoj Kumar; Kasarla Satish Reddy; Stefano Rinaldi; Bidare Divakarachari Parameshachari; Kavitha Arunachalam, Year: 2021, "A low area high speed FPGA implementation of AES architecture for cryptography application", Electronics, Vol: 10, No: 16, pp. 2023.
5. Vu Trung Duong Le; Hoai Luan Pham; Thi Hong Tran; Thi Sang Duong; Yasuhiko Nakashima, Year: 2023, "Efficient and high-speed cgra accelerator for cryptographic applications", In 2023 Eleventh International Symposium on Computing and Networking (CANDAR), pp. 189-195.
6. Jeevan Battini; Sivani Kosaraju, Year: 2023, "Design of efficient 22 nm, 20-FinFET full adder for low-power and high-speed arithmetic units", Silicon, Vol: 15, No: 2, pp. 993-1002.
7. Padmanabhan Balasubramanian; Nikos E. Mastorakis, Year: 2024, "Speed, Power and Area Optimized Monotonic Asynchronous Array Multipliers", Journal of Low Power Electronics and Applications, Vol: 14, No: 1.
8. Pratibhadevi Tapashetti; Dr Rajkumar B. Kulkarni; Dr SS Patil, "Implementation of High Speed MAC VLSI Architectures, Based on High Radix Modified Booth Algorithm", In International Conference on Advances In Electronics, Computer And Mathematical Sciences (ICIREMPS-2016) Sagar Institute of Research and Technology (SIRT) Bhopal, MP, India, Vol: 26.
9. Muteen Munawar; Zain Shabbir; Muhammad Akram, Year: 2023, "Area, delay, and energy-efficient full Dadda multiplier", Journal of Circuits, Systems and Computers, Vol: 32, No: 15, pp.2350258.
10. Imran Ahmed Khan, Year: 2022, "Design and implementation of carbon nano-tube based full adder at 32nm technology for high speed and power efficient arithmetic applications", In Journal of Physics: Conference Series, Vol: 2161, No: 1, pp. 012050.
11. Cedomir Marangunic; Felipe Cid; Andrés Rivera; José Uribe, Year: 2022, "Machine Learning Dependent Arithmetic Module Realization for High-Speed Computing", Journal of VLSI circuits and systems", Vol: 4, No: 01, pp. 42-51.
12. O. J. M. Smith; F. de Mendonça; K. N. Kantor; A. A. Zaky; G. F. Freire, Year: 2022, "Ultra low

potential operated fundamental arithmetic module design for high-throughput applications", Journal of VLSI circuits and systems, Vol: 4, No: 01, pp. 52-59.

13. Khamalesh Kumar Padmanabhan; Umadevi Seerengasamy; Abraham Sudharson Ponraj, Year: 2022, "High-speed grouping and decomposition multiplier for binary multiplication", Electronics, Vol: 11, No: 24, pp. 4202.