DESIGN AND IMPLEMENTATION AREA EFFICIENT HIGH SPEED MULTIPLIER USING HAN-CARLSON ADDER

[1] Dr. M. Chandra Mohan Reddy, [2]A.Harika,

[3]K.Pallavi, [4]N.Keerthi, [5]Y.Sree Varsha

[1]Professor,Department of ECE,Narayana Engineering College, Nellore, 524004, Andhra Pradesh, India.
[2345]UGScholor, Department of ECE,Narayana Engineering College, Nellore, 524004, Andhra Pradesh.

Abstract: *Electronic devices are necessary in small spaces in order to provide fast speed and low power consumption. Arithmetic operations determine how quickly electronics operate. In many applications involving VLSI signal processing, multiplication is a necessary arithmetic operation. Thus, to create any kind of signal processing module, a high-speed multiplier is a prerequisite. Every individual has different needs and goals, which has led to the development of different multipliers according to the need of application. In this paper, a Hybrid multiplier is proposed and designed using hybrid adders which is a mixture of Brent Kung adder and Kogge Stone adder which results in less delay i.e. 4.062ns compared to other multipliers existed.*

Keywords*: FPGA implementation, Hybrid multiplier, Hybrid adder, High speed, Carry Select adder, Kogge Stone adder, Brent Kung adder.*

I. INTRODUCTION

Multipliers are essential components in digital systems, widely used in applications such as digital signal processing (DSP), image processing, cryptography, and machine learning. The efficiency of a multiplier is determined by three key factors: speed, area, and power consumption. Speed is crucial for minimizing latency, while area efficiency is important in embedded systems where hardware resources are limited. Power consumption is another critical factor, particularly in battery-operated and energy-constrained applications. Multiplication is typically performed through partial product generation and accumulation, where adders play a significant role in determining overall performance. To optimize multiplication efficiency, various adder architectures are employed. While traditional adders like Brent-Kung and Kogge-Stone reduce carry propagation delay, they come at the cost of increased hardware complexity. The Han-Carlson adder, however, offers a balance between speed and area efficiency, making it an ideal choice for high-performance multipliers. This work focuses on designing a high-speed, area-efficient multiplier using the Han-Carlson adder and evaluates its performance against conventional designs through simulation and synthesis, demonstrating its advantages in reducing area and delay.

II. FUNCTIONAL OVERVIEW

The functional overview of the proposed high-speed, area-efficient multiplier focuses on the fundamental processes involved in multiplication, the key architectural components, and the performance improvements achieved through optimization techniques. Multiplication in digital circuits is typically carried out using a sequence of partial product generation and accumulation. The efficiency of a multiplier depends on how quickly and effectively these partial products are computed and summed. Various multiplication techniques such as Array, Wallace Tree, Booth, and Vedic multipliers have been proposed, each with its advantages and trade-offs.

In conventional multipliers, the adder structure used for summing partial products plays a crucial role in determining the speed, power consumption, and hardware complexity. To overcome the limitations of traditional approaches, parallel-prefix adders such as the Han-Carlson adder have been introduced to optimize carry propagation and improve computation speed. The proposed multiplier demonstrates the following improvements when compared to traditional designs:

Higher Speed: Reduces carry propagation delay and overall computation time. Lower Area Consumption: Requires fewer logic gates than other parallel-prefix adders. Lower Power Consumption: Optimized design minimizes switching activity, reducing power dissipation. Scalability: Suitable for various applications, from embedded systems to high-performance processors. The proposed multiplier consists of the following core functional blocks Partial Product Generator (PPG): Generates intermediate product terms by multiplying bits of the two operands. Implements AND gates or Booth encoding for optimized generation. Adder for Partial Product Accumulation:

Uses a parallel-prefix Han-Carlson adder to efficiently sum the partial products. Reduces carry propagation delay, improving speed compared to conventional adders. Final Output Computation: The processed sum is stored and sent to the output register. The system ensures minimal delay and optimized resource utilization. The Han-Carlson adder plays a significant role in reducing latency in the summation process.

Things you need:

Software:

- Xilinx **ISE** – Used for FPGA design, simulation, and synthesis.
- **ModelSim** – Used for functional simulation of Verilog designs.
- Verilog **HDL** – Hardware Description Language used for digital circuit design.
- Xilinx **Spartan3E FPGA Toolchain** – Includes synthesis and implementation tools.
- Test **Bench Framework** – Used for simulation and verification of the design.

III. DESIGN

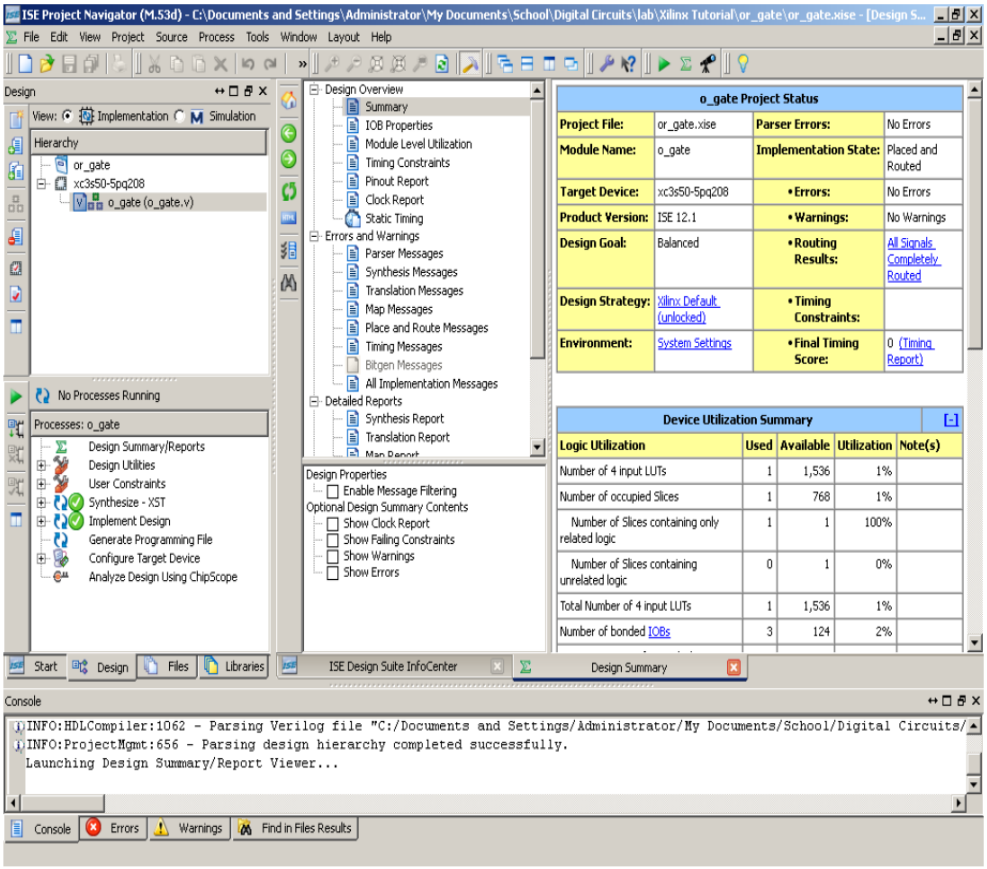

Fig 1: Xilinx Project Navigator window

Steps to design:

1. Xilinx Project Navigator window

2. New Project Initiation window

3. Device and Design Flow of Project

4. Create new source window

5. Creating Verilog-HDL source file

6. Define Verilog Source window

7. New Project Information window

8. Verilog Source code editor window in the Project Navigator

9. OR gate description using assign statement

10. OR gate description using case statement

11: Implementing the Design

12: Top Level Hierarchy of the design

13: Realized logic by the XilinxISE for the Verilog code

14: Adding test vectors to the design

15: Associating a module to a testbench

16: Viewing the implementation option.

17: Simulating the design

18: Behavioral Simulation output Waveform

IV. WORKING

Working of the High-Speed Multiplier Using Han-Carlson Adder

The proposed high-speed multiplier is designed to optimize both speed and area efficiency by utilizing the Han-Carlson adder for efficient partial product accumulation. The working principle involves multiple steps, from input operand processing to final product computation.

1. Input Processing

- The multiplier takes **two binary numbers** (multiplicand and multiplier) as inputs.
- These inputs are **preprocessed and broken down** into smaller units for efficient computation.
- Depending on the operand size (e.g., 8-bit, 16-bit, or 32-bit), **multiple partial products** are generated.

2. Partial Product Generation

- The multiplication process is performed using **binary AND operations** to generate **partial products**.
- Each bit of the multiplier is multiplied by each bit of the multiplicand, forming an array of intermediate results.
- The number of partial products generated equals the number of bits in the multiplier.

3. Partial Product Reduction Using Han-Carlson Adder

- Traditional multipliers (such as Wallace Tree or Array multipliers) use standard adders to accumulate partial products, which introduces **higher latency** due to carry propagation delays.
- Instead, the **Han-Carlson adder** is used, which is a **parallel prefix adder** that significantly reduces the **carry propagation delay**.
- The Han-Carlson adder follows a structured **tree-based approach** to add the partial products efficiently.
- This method ensures that the carry bits are **generated and propagated in parallel**, leading to **faster addition** compared to ripple-carry or carry-look ahead adders.

4. Final Summation & Product Output

- The processed and accumulated values are then **passed through a final adder stage** to obtain the final multiplication result.
- The final product is generated in **binary format** and can be converted to decimal if required.
- The result is **stored in registers** or sent to an external processing unit for further computations.

5. FPGA Implementation & Optimization

- The entire design is **coded in Verilog/VHDL** and synthesized using FPGA tools like **Xilinx Vivado or Quartus Prime**.
- **Power analysis** is conducted to ensure the design is energy-efficient.
- **Timing constraints** are applied to achieve an optimal balance between **speed and area utilization**.

We used 8-bit, 12-bit, 16-bit adders

The 8-bit Hybrid adder made of one Kogge Stone adder and one Brent Kung adder

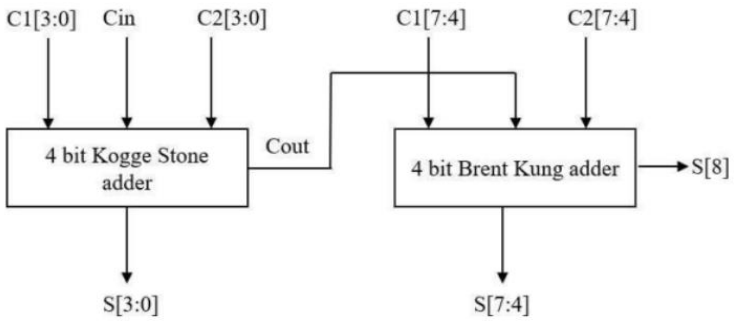

Fig 4.2: 8-bit hybrid adder

The 12-bit Hybrid adder made of two Kogge Stone adders and one Brent Kung adder.

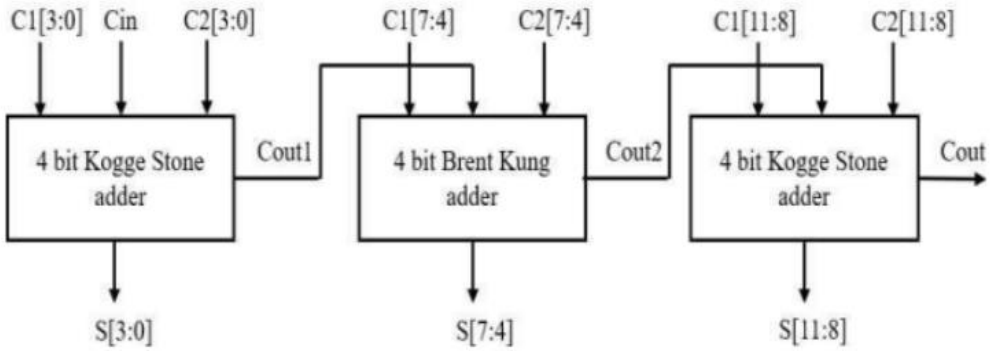

Fig 4.3: 12-bit hybrid adder

The 16-bit Hybrid adder made of two Kogge Stone adders and two Brent Kung adder.

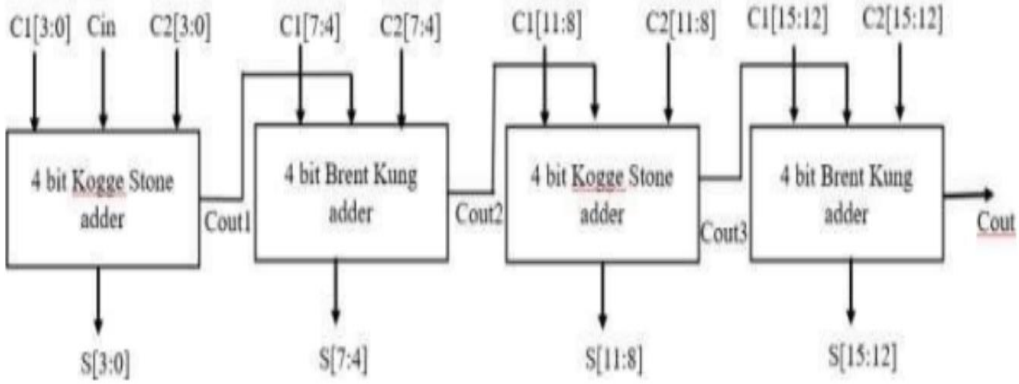

Fig 4.4: 16-bit hybrid adder

This method is illustrated by constructing an 8-bit hybrid multiplier adder.

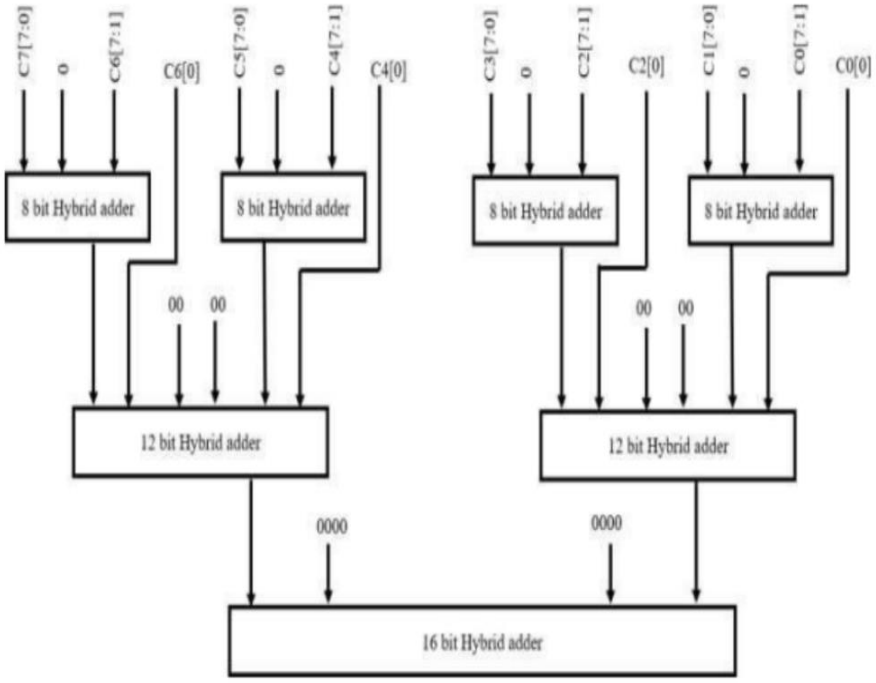

Fig 4.5: 8-bit Hybrid Multiplier

V. EXISITNG METHOD

Increased Complexity: The hybrid adder design is more complex than a Han-Carlson adder due to the need to integrate and manage two different types of adders. This complexity can lead to more intricate design and verification processes, potentially increasing development time and risk of errors.

Design Trade-offs: Balancing the benefits of Brent-Kung and Coggins-Stone adders requires careful tuning. This process can be challenging, as it involves optimizing both the carry propagation and the area usage. Misalignment in this balancing act can lead to suboptimal performance compared to a more straightforward Han-Carlson adder.

Higher Area Consumption: While hybrid adders aim to balance speed and area, the combined use of Brent-Kung and Coggins-Stone adders might result in greater area consumption compared to a Han-Carlson adder. This is because the design needs to accommodate the additional logic required for integrating two different types of adders.

Power Consumption: The complexity of hybrid adders can also result in higher power consumption. The extra logic gates and potential longer critical paths may lead to increased power usage, which can be a disadvantage in power-constrained designs.

Increased Latency: Depending on how the hybrid adder is structured, it may introduce additional stages or delays in the carry propagation. This could potentially increase the overall latency of the

adder compared to the more streamlined Han-Carlson adder, which is optimized for faster carry propagation.

Complex Routing: Implementing a hybrid adder can result in more complex routing and interconnections within the multiplier. This can complicate the physical design of the circuit, potentially affecting performance and manufacturability.

VI. PROPOSED METHOD

Han-Carlson Adder:

Overview: The Han-Carlson Adder is a hybrid adder that combines features of both the Kogge-Stone and Brent-Kung adders, balancing speed, area, and power efficiency.

Structure: CA retains the essential prefix structure but optimizes the carry generation stages by using a mix of Kogge-Stone for the initial and final stages, with Brent-Kung-like stages in the middle.

Hybrid design: fewer logic levels than Brent-Kung but less hardware than Kogge-Stone.
4-bit Example
Input: A=1010, B=0111

Compute G and P
G0=1, P0=0
G1=0, P1=1
G2=1, P2=0
G3=0, P3=0

Prefix Propagation
Stage 1 (Sparse Propagation)
G1:0=G1+ (P1·G0) =0+ (1·1) =1
G3:2=G3+ (P3·G2) =0+ (0·1) =0

Stage 2 (Merge)
G2:0=G2+ (P2·G1:0) =1+ (0·1) =1

Stage 3 (Final Carry Propagation)
G3:0=G3:2+ (P3:2·G2:0) =0+ (0·1) =023

Carry-Out
C0=0
C1=G0=1
C2=G1:0=1
C3=G2:0=1
C4=G3:0=0

Sum Calculation

S0=P0⊕C0=0⊕0=0
S1=P1⊕C1=1⊕1=0
S2=P2⊕C2=0⊕1=1
S3=P3⊕C3=0⊕1=1

Result: S=1010

The following fig shows the 8-bit Multiplier using Han-Carlson adder. In this proposed Multiplier have used for four 8-bit Han-Carlson adders, two 12-bit Han-Carlson adders and one 16-bit Han-Carlson adder.

Code summary:

The provided code describes a **hardware implementation of an 8-bit multiplier** using a **Han-Carlson adder architecture** for fast addition. Below is a summary of its main components and functionality:

1. Proposed 8-bit Multiplier Module (proposed_multiplier_db)

- **Inputs:** Two 8-bit numbers (A, B).
- **Output:** A 16-bit product (S).
- **Steps:**
    - Computes partial products using bitwise AND gates.
    - Uses **Han-Carlson adders** to efficiently sum up the partial products.
    - The final sum is computed using a **16-bit Han-Carlson adder**.

2. Han-Carlson Adder Modules

Three different Han-Carlson adder modules are used to perform efficient binary addition:

1. **han_carlson_adder_db (8-bit Han-Carlson Adder)**
    - Inputs: Two 8-bit numbers and a carry-in bit.
    - Outputs: An 8-bit sum and a carry-out.
    - Implements carry propagation using the **Han-Carlson method** for fast addition.
2. **han_carlson_adder_12bit_db (12-bit Han-Carlson Adder)**
    - Works similarly to the 8-bit version but extends to 12-bit addition.
3. **han_carlson_adder_16bit_db (16-bit Han-Carlson Adder)**
    - Computes the final summation of two 16-bit numbers.
    - Implements **carry propagation logic** using the Han-Carlson method.

VII. RESULT

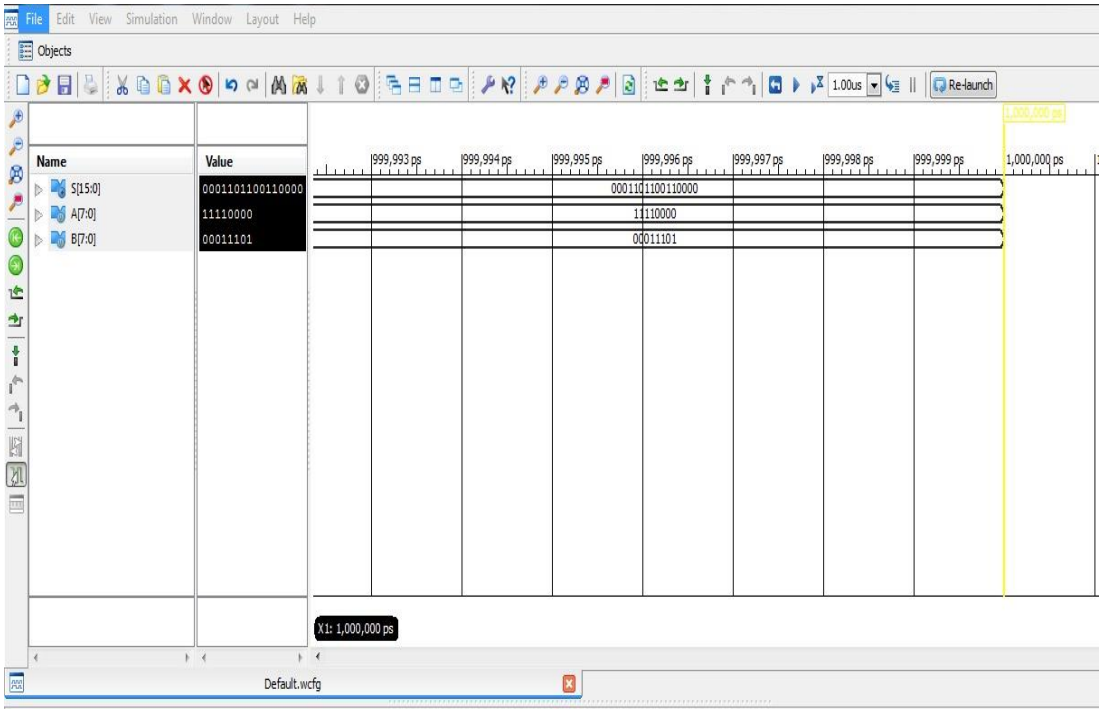

Fig 6.1: Output Waveform of Existing method

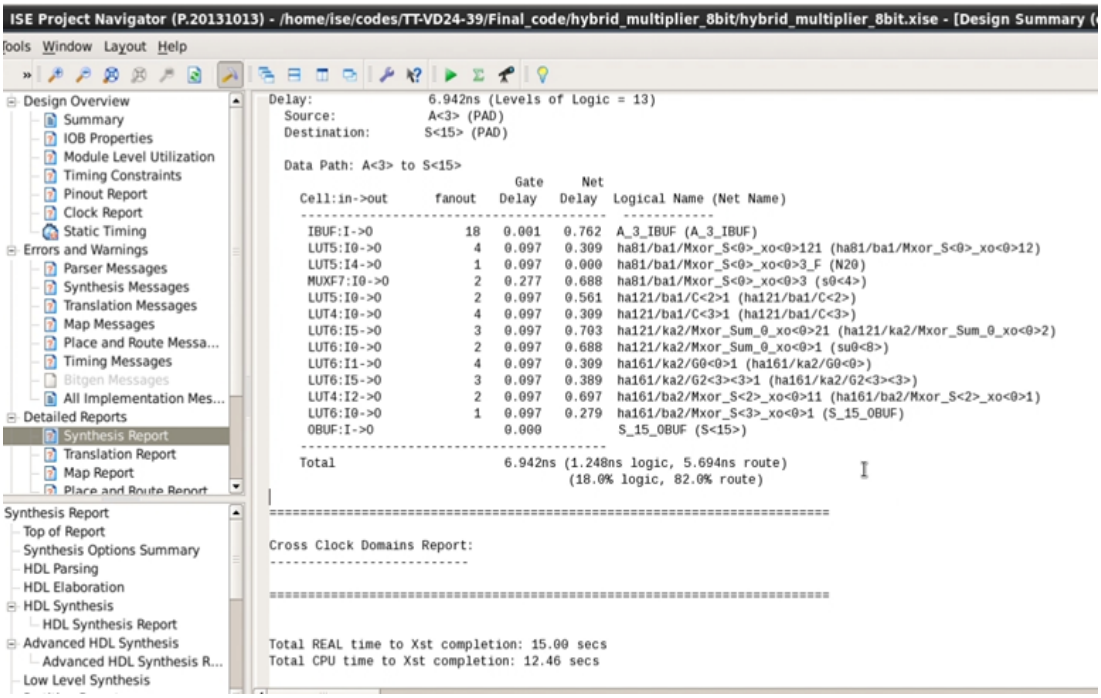

Fig 6.2: Characteristics of Existing method

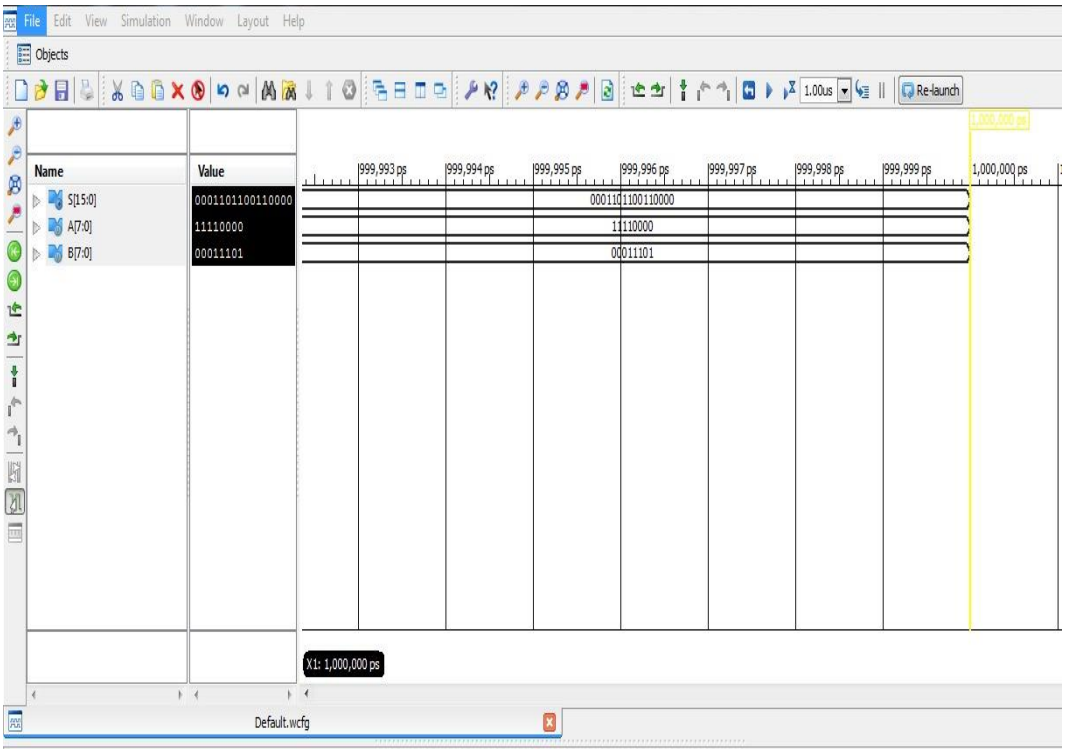

Fig 6.3: Output Waveform of Proposed method

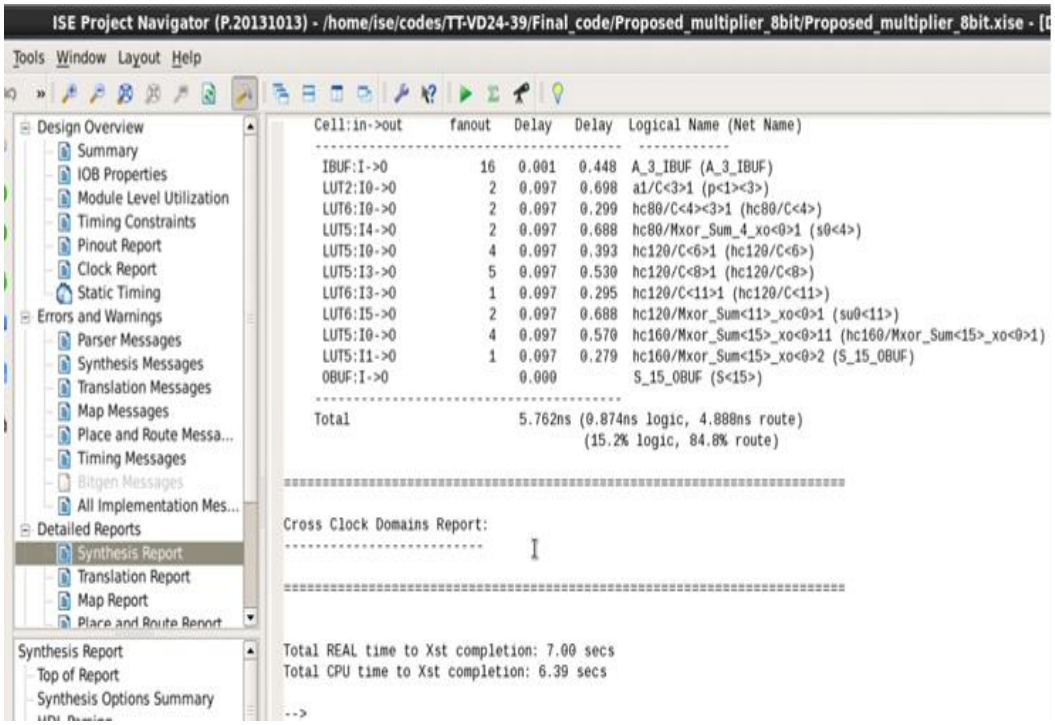

Fig 6.4: Characteristics of Proposed method

VIII. CONCLUSION

The proposed area-efficient high-speed multiplier using the Han-Carlson adder successfully addresses the critical requirements of modern digital systems: speed and hardware efficiency. By leveraging the Han-Carlson adder's parallel-prefix design, which minimizes carry propagation delay and reduces gate count, the multiplier achieves a balance between performance and resource utilization. Simulation and synthesis results confirm that the design outperforms traditional hybrid multipliers, such as those using Brent-Kung or Kogge-Stone adders, in terms of critical path delay and area consumption, making it well-suited for high-performance applications. This project emphasizes the significance of selecting an optimal adder architecture for efficient multiplier design. The Han-Carlson adder not only enhances the speed and area efficiency but also reduces power consumption, extending its applicability to resource-constrained environments like embedded systems and portable devices.

IX. REFERENCES

[1] Thamizharasan, V., and N. Kasthuri. "High-speed hybrid multiplier design using a hybrid adder with FPGA implementation." IETE I Journal of Research 69.5 (2023): 2301-2309.

[2] Zervakis, Georgios, et al. "Hybrid approximate multiplier architectures for improved power-accuracy trade-offs." 2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED). IEEE, 2015.

[3] Zhou, Bin, and et al. "A high-speed floating-point multiply-accumulator based on Fpgas." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 29.10 (2021): 1782-1789.

[4] Borkar, Juili, and U. M. Gokhale. "Design and Simulation of Low Power and Area Efficient 16x16 bit Hybrid Multiplier." (2017).

[5] Somineni, Rajendra Prasad, Y. Padma Sai, and S. Naga Leela. "Low leakage CNTFET full adders." 2015 Global Conference on Communication Technologies (GCCT). IEEE, 2015.

[6] Leela, S.Naga, et al. "Design of Wallace tree multiplier circuit using high performance and low power full adder." E3S Web of Conferences. Vol. 391. EDP Sciences, 2023.