FPGA Implementation of Power-Efficient Multiplier

Major Project report submitted in partial full-filment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

BY

MANAS JAIN (201017)

SAKSHAM (201004)

UNDER THE GUIDANCE

Prof. (Dr). SHRUTI JAIN

Department of Electronics and Communication Engineering Jaypee University of Information Technology Waknaghat, Solan-173234,

TABLE OF CONTENTS

DECLARATION

We at this moment declare that the work reported in the Bachelor of Technology Project Report entitled **"FPGA Implementation of Power Efficient Multiplier"** submitted at **Jaypee University of Information Technology, Waknaghat,** India is an authentic record of our work carried out under the supervision of **Prof (Dr.) Shruti Jain**. We have not submitted this work elsewhere for any other degree or diploma.

Manas Jain
201017

Saksham
201004

This is to certify that the above statement by the candidates is correct to the best of my knowledge.

Dr. Shruti Jain
Date:

Head of the Department/Project Coordinator

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT, P.O. – WAKNAGHAT,
TEHSIL – KANDAGHAT, DISTRICT – SOLAN (H.P.)
PIN – 173234 (INDIA) Phone Number- +91-1792-257999
(Established by H.P. State Legislature vide Act No. 14 of 2002)

JAYPEE EDUSPHERE
IGNITED MINDS INSPIRED SOULS

CERTIFICATE

This is to certify that the work reported in the B.Tech. project report entitled **"FPGA implementation of Power-Efficient Multiplier"**which is being submitted by **MANAS JAIN (201017) & SAKSHAM (201004)** in fulfillment for the award of Bachelor of Technology in **Electronics and Communication Engineering** by the **Jaypee University of Information Technology**, is the record of candidate's work carried out by him under my supervision. This original work has not been submitted partially or fully anywhere else for any other degree or diploma.

Dr. Shruti Jain
Associate Dean (Innovation)
Professor, Department of Electronics & Communication Engineering
Jaypee University of Information Technology, Waknaghat

ACKNOWLEDGEMENT

I am really grateful and wish my profound indebtedness to the supervisor Dr. Shruti Jain, Professor and Associate Dean (Innovation), Jaypee University of Information Technology. Deep knowledge & keen interest of my supervisor in the field of "VLSI" to carry out this project. Her endless patience, scholarly guidance, encouragement, energetic supervision, valuable advice, reading many inferior drafts, and correcting them at all stages made it possible to complete this project.

I want to express my heartiest gratitude To Dr. Rajiv Kumar, H.O.D Department of Electronics and Communication Engineering for his kind help in finishing my project.

Finally, I must acknowledge with due respect the constant support and patience of my parents.

Manas Jain (201017)

Saksham(201004)

LIST OF ABBREVIATIONS

1. VLSI = Very Large-Scale Industry
2. DSP = Digital Signal Processing
3. XSG = Xilinx System Generator
4. FPGA = Field Programmable Gate Arrays
5. HA = Half Adder
6. FA = Full Adder
7. MUL. = Multiplier
8. IC = Integrated Circuits
9. DIP = Dual In-Line Packaged
10. PGA = Pin Grid Array
11. BGAs= Ball Grid Arrays
12. CPU = Central Processing Unit
13. ROM = Read Only Memory
14. RAM = Random Access Memory
15. UDP =User-defined primitive
16. S = Sum
17. C= Carry
18. SD = SC
19. WTM= Wallace Tree Multiplier
20. ASIC = Application-Specific Integrated Circuit
21. MOS = Metal Oxide Semiconductor
22. HDL = Hardware Descriptive Language
23. RTL = Register Transfer Level
24. CPLD = Complex Programmable Logic Device
25. CLB = Configurable Logic Blocks

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

One of the main issues with digital design across a wide range of applications, from data centres to embedded and battery-powered devices, is the creation of energy-efficient circuits. Many different techniques have been developed in the past few years to boost speed while lowering power consumption, and there are some intriguing potential benefits. The basic tenet of this strategy is that significant savings in hardware design metrics like key path delay, design space, and power consumption can be achieved by lowering the output's accuracy requirement. This is the fundamental idea that guides such a strategy. Conversely, this paradigm can only be applied in situations where a preexisting tolerance for small and insignificant errors has been established. The domains of media processing, machine learning, and data mining are a few instances of possible uses. It's interesting to observe that the number of applications that primarily rely on data and machine learning has increased recently. Multiplier is growing in popularity for image/signal processing applications. In the era of real-time applications, multipliers are becoming more and more important. Adders are essential to the design and operation of digital circuitry and signal processing applications in a wide range of computer types, including microprocessors and digital signal processors (DSPs). The main problem that needs to be solved when adding is the propagation of delay in the carry chain. As the bit-width of the input operands increases, the carry chain length grows and becomes more stretched out. When it comes to Very Large-Scale Integration (VLSI), the best adder designs are classified as having a Parallel. A half adder and a full adder are the essential building blocks for adding the final sum in the final addition step of each of the three Wallace tree multiplier designs. A range of compressors must be employed for the installation of the energy-efficient approximate multiplier blocks; VIVADO 18.1 Design Tool is used for all simulations. Furthermore, the suggested adders

CHAPTER 1

INTRODUCTION

We chose to create multipliers using various adders since they are utilized in many different engineering domains. VLSI is the process by which an integrated circuit (IC) is constructed on a single chip from millions or billions of MOS transistors. VLSI got its start in the 1970s with the creation and broad application of MOS (Metal Oxide Semiconductor) integrated circuit chips, which made advanced semiconductor and telecommunication technologies possible. CPU and memory chips are designed using VLSI devices. The fundamental goal of VLSI technology development is to make ICs capable of performing a limited number of tasks. An electronic circuit may contain a CPU, ROM, RAM, and other glue logic components. All of them could be combined into one by IC designers.

Design Flow: -The technique in which we build an integrated circuit (IC) using millions or billions of MOS transistors on a single chip is known as VLSI. VLSI circuits are used, such as in personal computers, graphic cards, digital cameras, camcorders, cell phones, embedded processors, anti-lock braking systems in cars, personal entertainment systems, medical electronic systems, etc. The requirements of today's electronic gadgets and sysms are ideally suited to VLSI technology. VLSI technology must be growing to drive advancements in electronics due to the rising demand for smaller, more compact, reliable, and functioning devices. The number of jobs is increasing in the VLSI design as India develops.

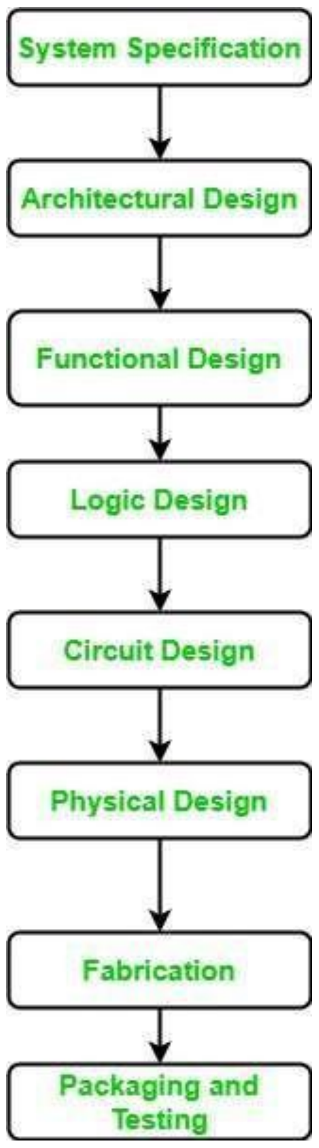


Figure 1.1 VLSI Design Flow [1]

System Generator: - The goal of the intended final product is written in this step. System specification should include considerations for architecture, performance, and how the system will communicate with the external environment.

Architectural Design: - Aspects of architectural design include the integration of analogue and mixed-signal blocks, memory management, internal and external connectivity, power requirements, process technology, and layer stack selection.

Functional Design: - The primary goal of this is to create a high-performance architectural design while adhering to the specification's cost constraints.
Logic Design: - The structure of the desired design complements the behavioural representation of the desired design. Testability, performance improvement, and logic minimization are the three fundamental criteria for logic design.

Circuit Design: - The logic building blocks of the intended design are replaced by electronic circuits, which are composed of electronic components like resistors, capacitors, and transistors.

Physical Design: - The intended system's real layout has been completed, showing where each component will go in the circuit and how they are all connected.

Fabrication: - The intended design is sent to be manufactured once it has been confirmed and actually laid out. The term "tape out" describes the process of transferring the intended design to the production line. The process of producing the data required for manufacturing is referred to as "streaming out."

Packaging and Testing: - Once the desired design is manufactured, functional chips are packaged. Packaging is configured early in the intended design process, along with the application, cost, and form factor criteria. A few possible package types are Ball Grid Arrays (BGAs), Pin Grid Arrays (PGAs), and Dual In-Line Packaged (DIPs).

MOTIVATION

In the computing system, the consumption of energy is an avital parameter. Nowadays, everyone needs high computational speed with low power consumption devices. Emerging high-performance applications require increasingly big datasets to be processed with high efficiency.

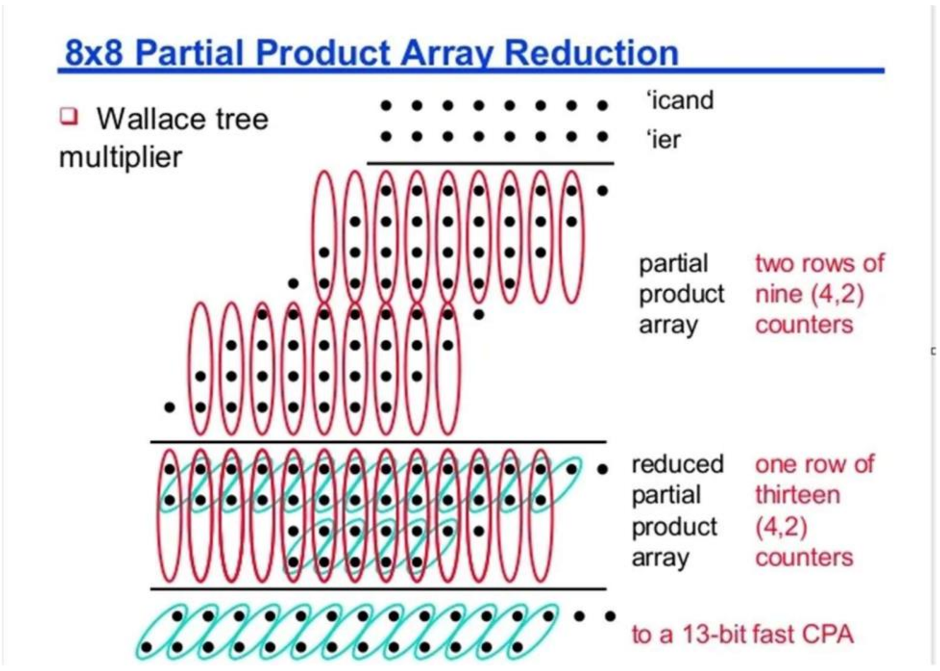
With the growing demands of electronic devices, important changes occurred in the electronic community, especially in the VLSI world. In the VLSI world, digital circuits play a vital role and are highly complex as they consist of a large number of transistors and logic gates at their most detailed level. Gordon Moore in 1965 states that every year the transistor count on integrated circuit (IC) doubles. But later on, Moore observed that not every year but about every two years the transistor count on IC doubles. This observation is known as Moore's Law. According to Moore's Law, the processing speed of the device increases as the transistor count doubles in every 2 years due to improvements in technology. As a consequence of shrinking dimensions, Robert Dennard 1974 observed a few things with every technology generation power consumption remains the same if transistor density doubles. Dennard scaling law is formulated after his observation which states that power density remains constant as the transistor size gets smaller. When the computation performance of Moore's law (transistors double approximately every 2 years) is combined with the Dennard scaling then the transistor doubles about every 1.5 years and the performance per watt even grows faster. In around 2005-2007, there is a breakdown of Dennard scaling. The main reason for the breakdown in Dennard scaling is the risk of thermal runaway [2].Multiplier: -In digital signal processing (DSP) applications we mainly used multipliers. Out of all the arithmetic operations now in use, the multiplier is the most widely used. It is employed to multiply two numbers using a variety of methods. To form an efficient multiplier our main focus is on the four aspects, i.e., speed, power consumption, area, and accuracy. We are trying to cover this in all existing popular Wallace tree multipliers. Finally, we compared the performance characteristics of each multiplier, including speed, time delay, area, complexity, accuracy, and power consumption [3].

Types Of Multipliers: -

- Array
- Wallace Tree
- Vedic
- Parallel
- Serial
- Approximate
- Logarithmic
- Sequential

WALLACE TREE MULTIPLIER

It takes less time for accumulation in this multiplier because the generated partial products are added in parallel. The partial product reduction of the 8 x 8 Wallace multiplier is displayed in Figure 1.2. The blue dotted circle in Figure 4.2 represents the half adder, and the red dotted circle the full adder. However, the Wallace multiplier is unique from other multipliers, such as array multipliers, in that it adds up all partial products. The Wallace tree multiplier has multiple stages where all partial products are added. The parallel addition of three rows is considered at every level, which facilitates and expedites the process. Any group that has less than three rows is kept the same and moves directly on to the next addition step.



8x8 Partial Product Array Reduction
Wallace tree multiplier
'icand
'ier
partial product array
two rows of nine (4,2) counters
reduced partial product array
one row of thirteen (4,2) counters
to a 13-bit fast CPA

Figure 1.2: - WTM**[4]**

Application: Microprocessors in personal computers, graphic card chips, digital camera or camcorder chips, cell phone chips, embedded processors, anti-lock braking systems for cars, personal entertainment systems, medical electronic systems, etc. are all examples of devices that use VLSI circuits.

ARRAY MULTIPLIER

The items in the conventional array multiplier are added using the carry-save addition method. Depending on how the carry save adding mechanism works, the first row will either have full adders or half adders. If complete adders are used in the first row of the partial products, then Cin will be taken to be zero.

To generate partial products and test each bit of the multiplier one at a time, a sequence of add-and-shift micro-operations is required. Two binary values can be multiplied in a single micro-operation using a combinational circuit that generates the product bits concurrently. This method multiplies two values quickly because it only takes the signals a short time to pass through the gates that make up the multiplication array. However, prior to the invention of integrated circuits, an array multiplier was not cost-effective due to its high gate count requirement.

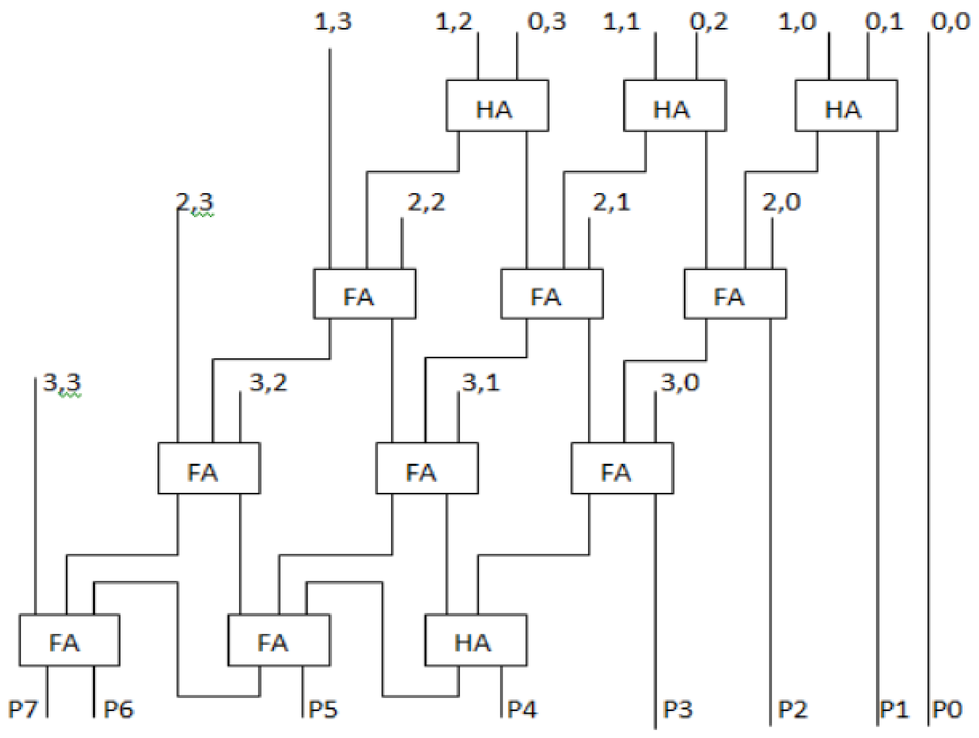


Figure 1.3: - Array multiplier

Application: Arithmetic operations such as filtering, Fourier transformation, and image coding are carried out using array multipliers.

VEDIC MULTIPLIER

The Urdhva Tiryakbhyam Sutra was utilized in this. In Urdhva Tiryakbyham Sutra, "Vertically and Crosswise" is the actual meaning. This sutra applies to all multiplication situations. The partial products and their total are computed simultaneously using this method. Figure 1.4 illustrates the procedures for multiplying two numbers by two using the Urdhva Tiryakbyham Sutra. As seen in Figure 1.5, the same process can be expanded to include 4x4 multiplication.

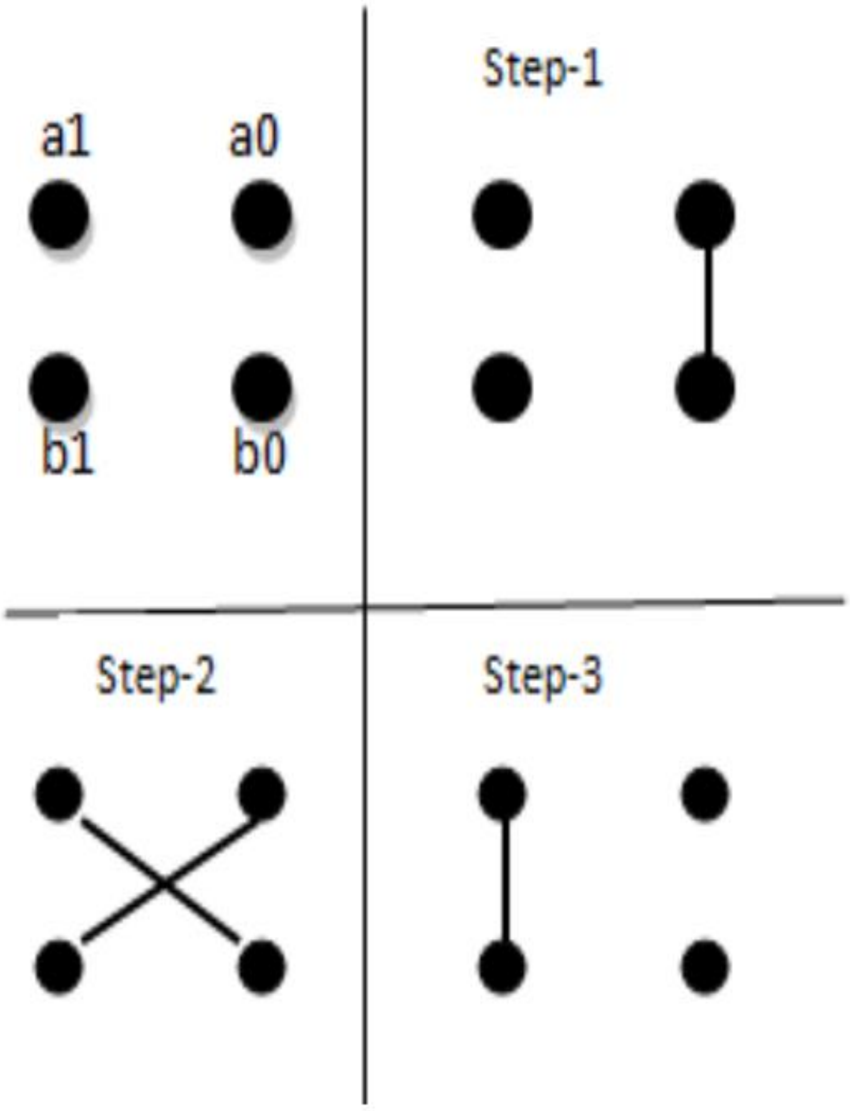


Figure 1.4: - Using the Urdhva Sutra to multiply two by two.

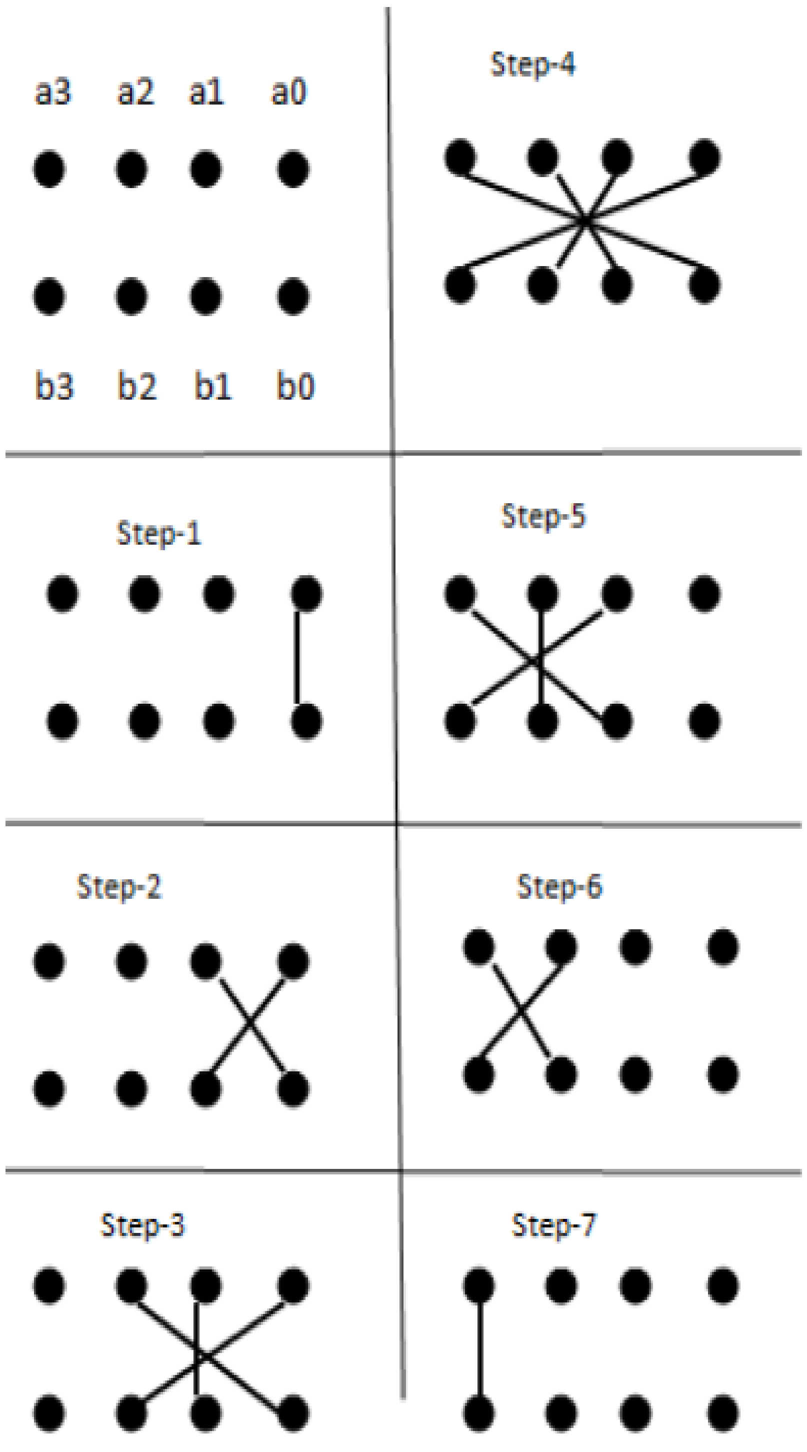


Figure 1.5: - Using the Urdhva Sutra to multiply 4 by 4.

The 2-bit multiplier serves as the foundation for higher order multipliers like 4bits, 8bits, and 16bits multiplication.

SOFTWARE AND HARDWARE USED

VIVADO 18.1

The software package Xilinx With new features for high-level synthesis and system-on-a-chip development, Vivado Design Suite takes the place of Xilinx ISE. Designs written in Hardware Description Language (HDL) are synthesised and analysed using it. Using Vivado, the design flow has been entirely revised and redesigned.

VERILOG

The most advanced technique for creating digital and computer systems is the Verilog hardware description language. A digital system can be modeled using the C-like language Verilog HDL, which also has some Pascal syntax. Verilog is simple to learn since it uses a syntax that combines C and Pascal. The most popular HDL on the market is Verilog. Combinational and sequential logic, as well as storage devices that are level-sensitive and edge-triggered, can all be described using the Verilog HDL. Any ASIC, FPGA, or CPLD design may be swiftly prototyped and debugged using the VIVADO simulation. It is a user-friendly environment that shows all of the ports and variables from a module to a logic gate. Algorithms, Boolean equations, and specific logic gates can all be modeled using Verilog.

The Verilog language includes logic primitives such as AND, OR, NAND, and NOR gates. A user-defined primitive (UDP), which can be instantiated into a module similarly to a built-in primitive, is another option available to designers UDPs can be any logic function such as a multiplexer, decoder, encoder, or flip-flop. Three main modeling structures, namely dataflow, behavioral, and structural, can be used to represent designs. It is also possible to develop modules using a mixed-design approach, which combines the aforementioned constructs with native and user-defined primitives.

FPGA: A field programmable gate array (FPGA) is a fully fabricated IC chip in which the interconnections can be programmed to implement different functions. Thousands of logic gates on an FPGA device must be coupled to implement any logic function. It has three main components as shown in Fig 1.3.

- Array of configurable logic blocks (CLBs)
- Programmable interconnects
- I/O buffer

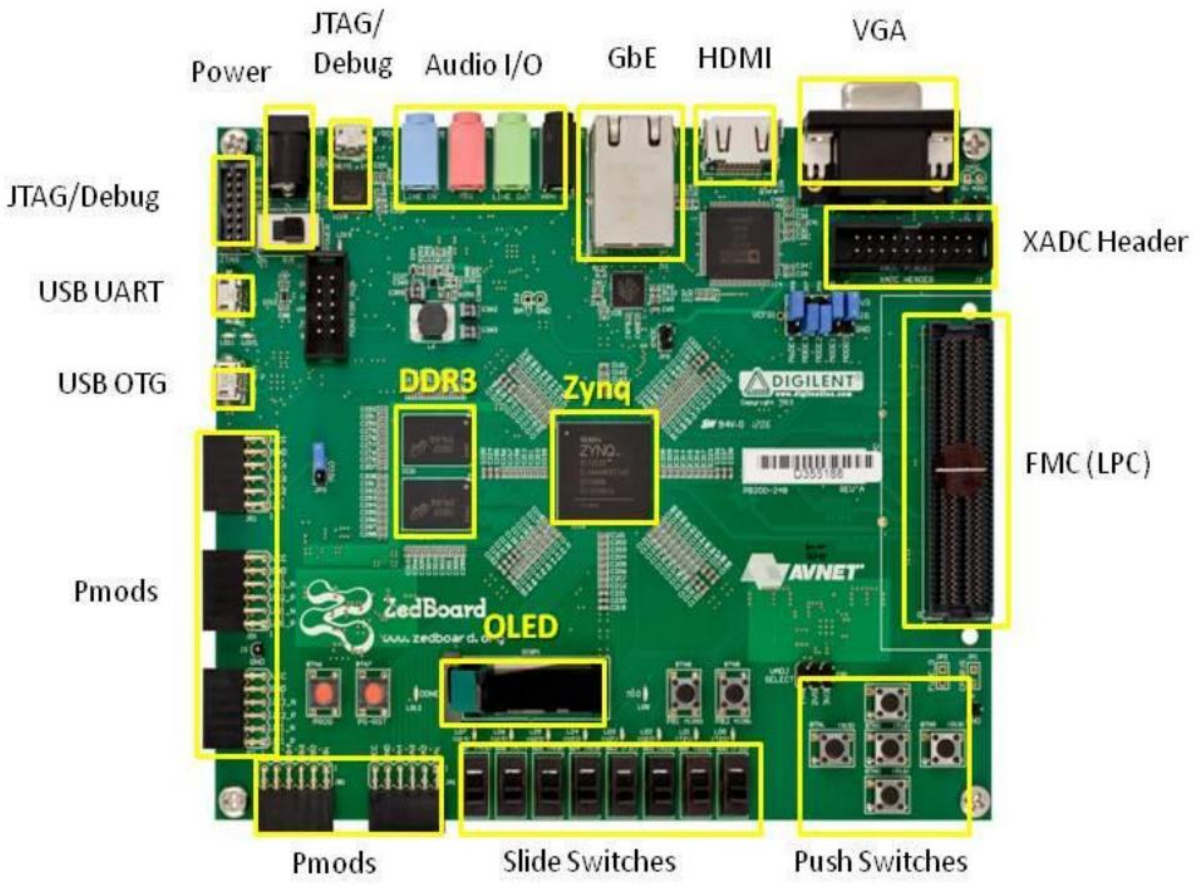


Figure 1.6: - Zed Board FPGA Board

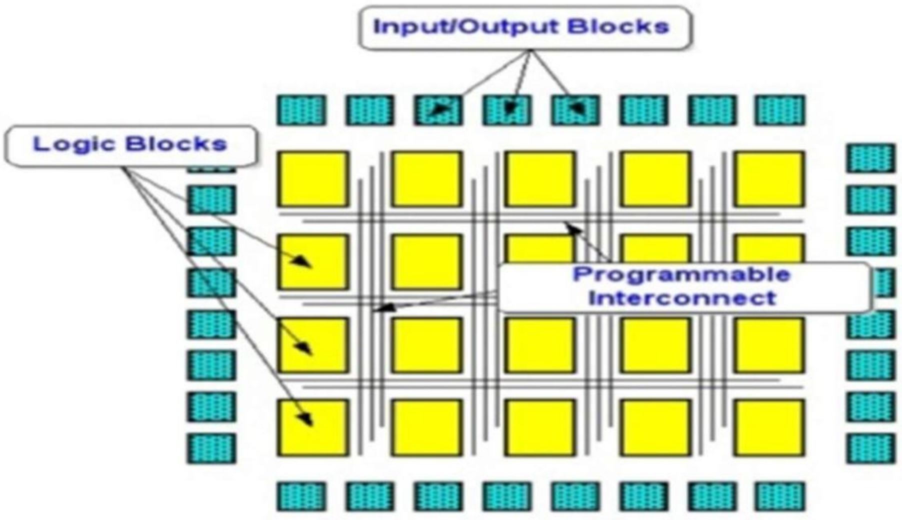


Figure 1.7: - CLBs **[5]**

In the FPGA-based design, the functionality of the design is initially described in a behavioral netlist. Hardware description languages like Verilog or VHDL are used for this. The gate-level design is created once the netlist has been synthesized.

The logic blocks are then mapped into the accessible logic cells as the next step. The mapping of technology is the name of this technique. Placement and routing come next, configuring the CLBS and defining linkages. The next stage is to create the bit stream and, using a software interface, download it into an FPGA chip. The FPGA chip can thereafter perform the necessary function so long as the power is ON or once it has been reprogrammed. The use of an FPGA-based design has the advantage of having a very quick design cycle, which makes it ideal for developing low-cost, low-volume prototypes. A concept can be swiftly implemented in hardware to determine whether it is worthwhile to implement in a significant number of instances. However, performance is not optimized in FPGA-based designs. An FPGA-based design typically takes a few hours to a few days to complete.

Logic Block Organization: Figure 1.4 shows the pin positions for the FPGA logic block in which the output pin is connected to routing wires in which both of the channels have the right and lower logic block, however, each input can only be accessed from one side of the logic block.

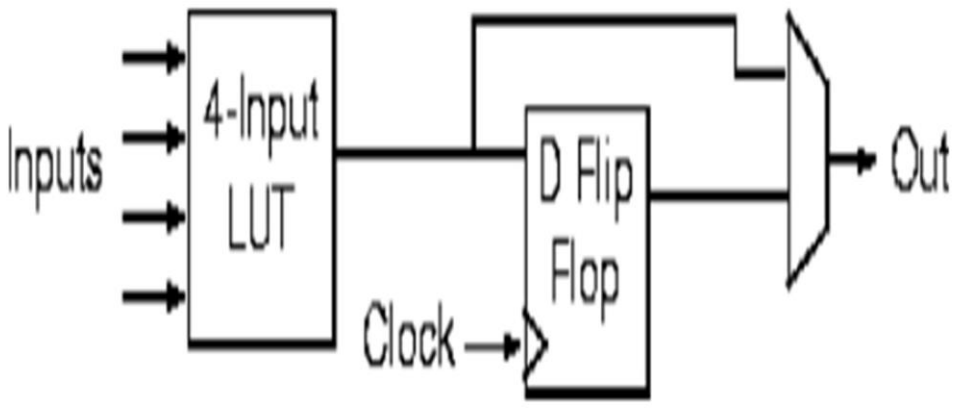


Figure 1.8: - Logic Block **[5]**

Locations of Logic Block Pins: Each input pin of a logic block can be wired to any of the adjacent wiring segments in the channel. Any wiring segment in the channels next to a logic block output pin can be connected to it. Thus, using the standard FPGA notation, Fc = W (tracks per channel). The situation should be evident from the logic block pins as shown in Figure 1.5.

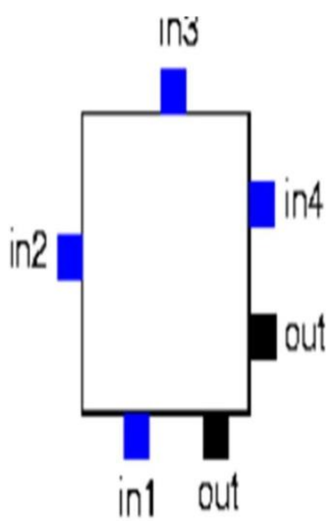


Figure 1.9: - Logic Block Pins **[5]**

The power-efficient multiplier will be simulated using VIVADO18.1.

CHAPTER -2

LITERATURE REVIEW

Chatterjee *et al.* (2022) [6] suggest dividing an unsigned approximate multiplier architecture into three parts. To increase hardware savings without sacrificing accuracy, the least significant part, or the part that contributes the least to the partial product (PP), is replaced with a new constant compensation term. A new 4:2 approximate compressor simplifies the PPs in the middle section, and an effective yet basic error correction module corrects the approximation error. Since estimating this part of the multiplier will lead to a significant error, it is implemented with exact logic. The experimental results of the 8-bit multiplier show reductions in power and power-delay products of up to 47.7% and 55.2%, respectively, compared with the exact design's 36.9% and 39.5%.

Ravindran *et al.*(2022)[7] Create an app for edge detection using an approximation multiplier. This work is based on the idea that several applications can tolerate a certain level of approximation in the design, and that approximation can significantly reduce the device's required speed, area, and time. In this work, the multiplier is modeled based on an approximate VERILOG (HDL) computation using Xilinx. For approximation purposes, the multiplier must therefore meet better area, power, and timing requirements. As a specific application, image processing is the main focus of the design. Advantages over conventional multipliers in terms of areas, power, and time are highlighted in the design. The suggested multiplier is modeled using Modelsim, synthesized using Xilinx Vivado, and generated using Verilog HDL.

Sabetzadeh *et al.(*2023) [8] This brief introduces an extremely effective error-correcting approximation multiplier. The least significant half of the product is handled as a constant compensating term by the suggested multiplier. The second half is meticulously calculated to yield an extremely high hardware-accuracy trade-off. Furthermore, a simple-yet-efficient error compensation module (ECM) is introduced, which greatly improves accuracy. HSPICE is used to simulate the proposed multiplier, which uses 7nm tri-gate FinFET technology. There is an average improvement when comparing the suggested design

Strollo *et al.*(2022) [9] Apply static segmentation to analyse multiplier approximations. In these circuits, each of the two n-bit operands generates a segment of m bits or a set of m contiguous bits. The two segments are fed into a tiny m×m internal multiplier, which modifies its output accordingly to yield the required result. We study signed and unsigned multipliers, and we suggest a novel segmentation strategy for the latter. We also offer straightforward and efficient correction methods that, with less expensive hardware, can significantly lower the approximation error.

Neyaz*et al.*(2020)[10] carries out various 4-bit and 8-bit multiplier types. Multipliers are widely used and even necessary components of many signal processing applications in the modern digital world. All of the various 4- and 8-bit multipliers are implemented and compared in this essay. To compare the designs, the implementation makes use of both the FPGA and the ASIC. The FPGA is implemented using the Xilinx Artix-7 FPGA-equipped NEXYS 4 DDR. The results demonstrate that the Wallace tree has the lowest power delay product in an ASIC implementation and the lowest delay in an FPGA implementation, respectively. The FPGA and ASIC implementations of the Baugh Wooley architecture take up the least amount of space.

Moaiyeri *et al.* (2022) [11] Proposals for energy- and quality-efficient approximate multipliers are presented, based on novel approximation compressors. To minimise the number of transistors, we generate the complemented partial products using NAND gates. New approximation compressors are also built with different performance and accuracy levels. As a result, three hybrid approximate multipliers that offer various hardware efficiency and accuracy trade-offs are put forth. The suggested architectures are simulated using the 7nm FinFET model, a contemporary technology, using HSPICE. Furthermore, MATLAB is used to assess the approximation multipliers' performance in neural network and image processing applications.

Immareddy *et al.* (2022)[12] Multiplication is one of the basic operations in all digital systems. New challenges have emerged in the development of VLSI (Very Large-Scale Integration) as a outcome of the digital systems assessment. In digital signal processing, multipliers are typically utilised. These days, the advancements in technology are driving up demand for this. Many different multiplier designs have been developed to increase its speed. The numerous investigations that have been conducted since 2015 are examined in this document. This article looks at studies that employ various multipliers. Prior to comparing the adders, obtain the Ripple. A Carry Look Forward Adder is in the middle of the spectrum and effectively balances the difficulties presented by time and space. Multipliers came into focus after adders were created and assessed. Wallace Tree Multiplier was first selected, followed by Parallel Multiplier. In the meantime, it was found that the latency was greatly decreased by Wallace Tree applications that used Carry Save Adders. Research manuscripts are compared and analysed using a variety of criteria in this review. An overview of our current knowledge of these multipliers can be found in this publication. In this comparative analysis, timeline plays a part.

Ram *et al.* (2022)[13] This work provides a Han-Carlson adder and an FPGA implementation of a modified Wallace multiplier. High-quality multipliers are essential because they are the main components of microprocessing circuits and signal processing boards. One way to create delay-efficient multipliers is to modify a suitable adder to accommodate the addition of partial products. To improve the adder's features, like power levels and area usability, the traditional Wallace tree multiplier must use the Han-Carlson adder instead of the ripple carry adder. With the aid of the XILINX programme, the enhanced Wallace multiplier is simulated and coded in Verilog HDL. Using the SPARTAN 3E FPGA kit, the multiplier's performance is confirmed.

Toan *et al.* (2020)[14] demonstrates how to use recently proposed approximation logic compressors at various accuracy levels to efficiently create approximate multipliers on Field Programmable Gate Arrays (FPGAs). In comparison with the state-of-the-art works, our approximate multiplier designs provide higher power-delay-area products (PDAP) increases at comparable accuracies. Furthermore, our solutions outperform the Lookup Table based multiplier intellectual properties available on an FPGA in terms of delay, occupied area, and dynamic power dissipation. Specifically, we can achieve up to 7.1 x, 8.3 x, and 5.0 x PDAP increases with our proposed 8-, 16-, and 32-bit multipliers.

LIST OF LITERATURE REVIEW

Serial No.	Year	Work	Remarks
1.	2022[6]	How does Wallace tree methods Work?	Implemented the Wallace tree method.
2.	2022[7]	How edge detection is calculated and detect in multiplier?	Implement an Approximate Multiplier with Edge Detection Application.
3.	2023[8]	Calculation of Power Of multiplier.	Implemented the Power Efficient Approximate Multipliers.
4.	2022[9]	Make Wallace multiplier using less number of logic resources.	Implemented the multiplier which requires less logic resources but produces some error.
5.	2020[10]	Different Bits implementation of Multipliers.	Implements the different types of 4-bit and 8-bit multipliers.
6.	2022[11]	Matrix Vector Importance.	Implements the matrix vector multiplier.
7.	2022[12]	Multiplier Works	Implements the multiplier.
8.	2022 [13]	FGPA implementation.	FPGA implementation of Wallace multiplier with the help of Han-Carlson adder.
9.	2020 [14]	Multi-Level Approximate Multiplier.	FPGA implementation of Multi Level Approximate Multiplier.

CHAPTER 3

METHODOLOGY

In this project, various multipliers like Array, Wallace Tree, and Vedic multipliers are designed. The implementation for **2×2, 4×4, 8×8,16×16** bits was done using Verilog coding in VIVADO. The various parameters such as a number of slices, number of IOB, and power were compared for different multipliers. The designs are optimized for higher speed, less power, and fewer numbers of IOB & Slice which is the main requirement for the Image / Signal processing. The designs were further implemented on the FPGAs platform using the bitstream. The results of various multipliers like Vedic, Array, and Wallace tree were compared based on different performance parameters and different bits. We are moving on with the applications of the best multiplier among them.

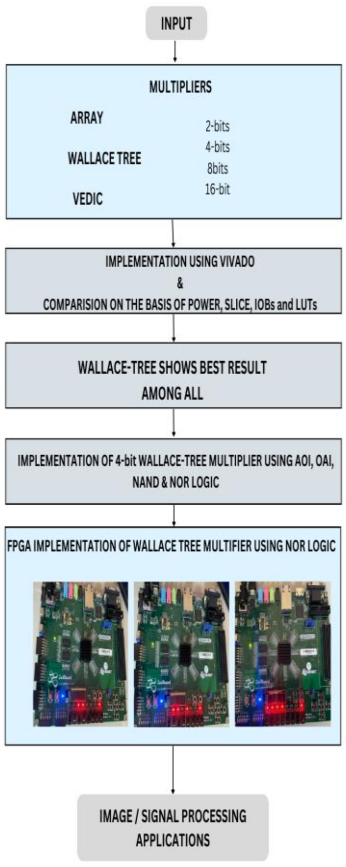

INPUT
MULTIPLIERS
ARRAY
WALLACE TREE
VEDIC
2-bits
4-bits
8bits
16-bit
IMPLEMENTATION USING VIVADO
&
COMPARISION ON THE BASIS OF POWER, SLICE, IOBs and LUTs
WALLACE-TREE SHOWS BEST RESULT AMONG ALL
IMPLEMENTATION OF 4-bit WALLACE-TREE MULTIPLIER USING AOI, OAI, NAND & NOR LOGIC
FPGA IMPLEMENTATION OF WALLACE TREE MULTIFIER USING NOR LOGIC
IMAGE / SIGNAL PROCESSING APPLICATIONS

Figure 3.1: - Methodology

ARRAY MULTIPLIER

In the traditional array multiplier, the items are added via carry save addition. The first row will either contain half adders or full adders based on the carry save adding mechanism. Cin will be assumed to be zero if complete adders are used in the first row of the partial products.

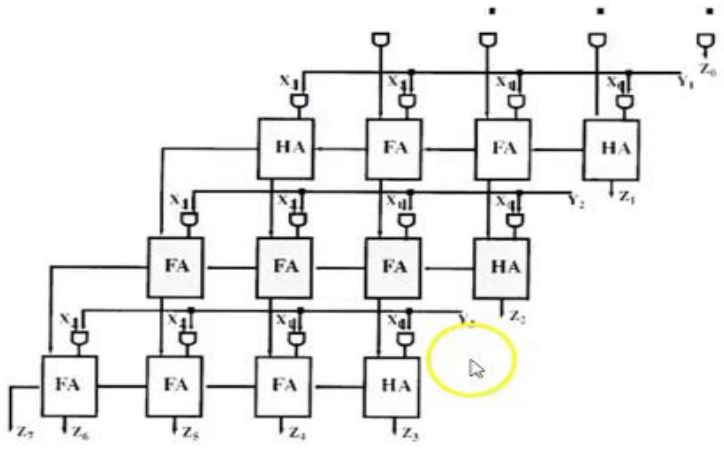


Figure 3.2: -Circuit Diagram of Array Multiplier

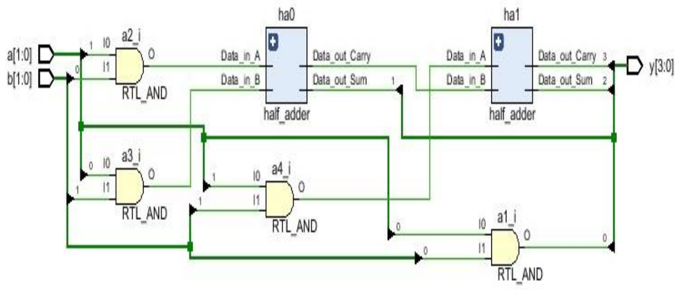
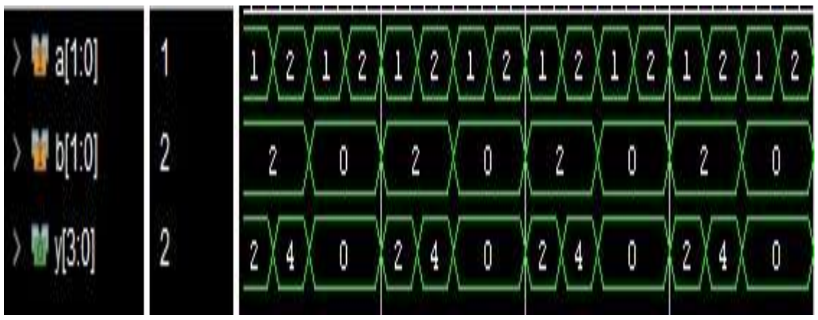


Figure 3.3: -SC of 2bits Array Multiplier



```
a[1:0]   1   1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
b[1:0]   2   2   0   2   0   2   0   2   0
y[3:0]   2   2 4 0   2 4 0   2 4 0   2 4 0
```

Figure 3.4: - Waveform of 2bits Array Multiplier

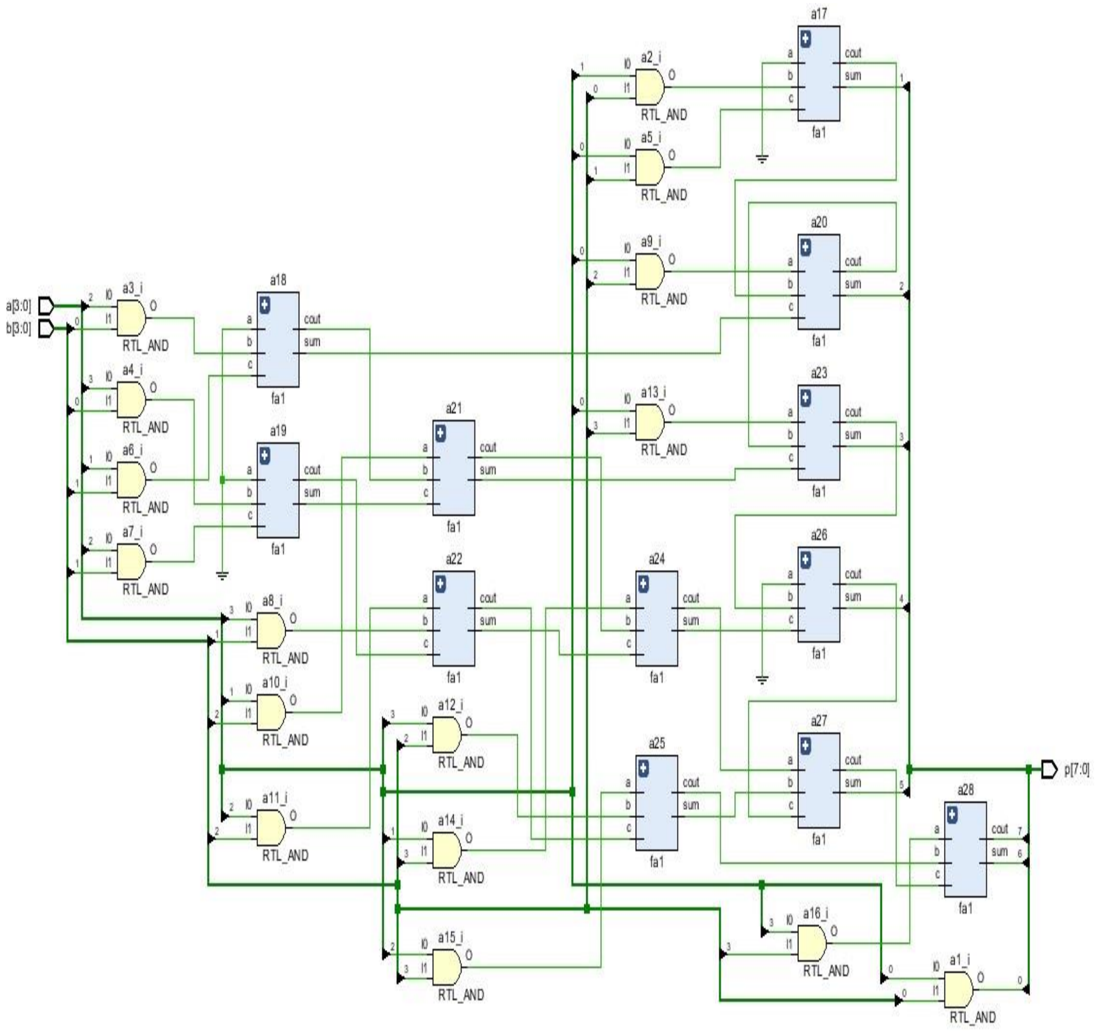


Figure 3.5: **-** SC of 4bits Array Multiplier

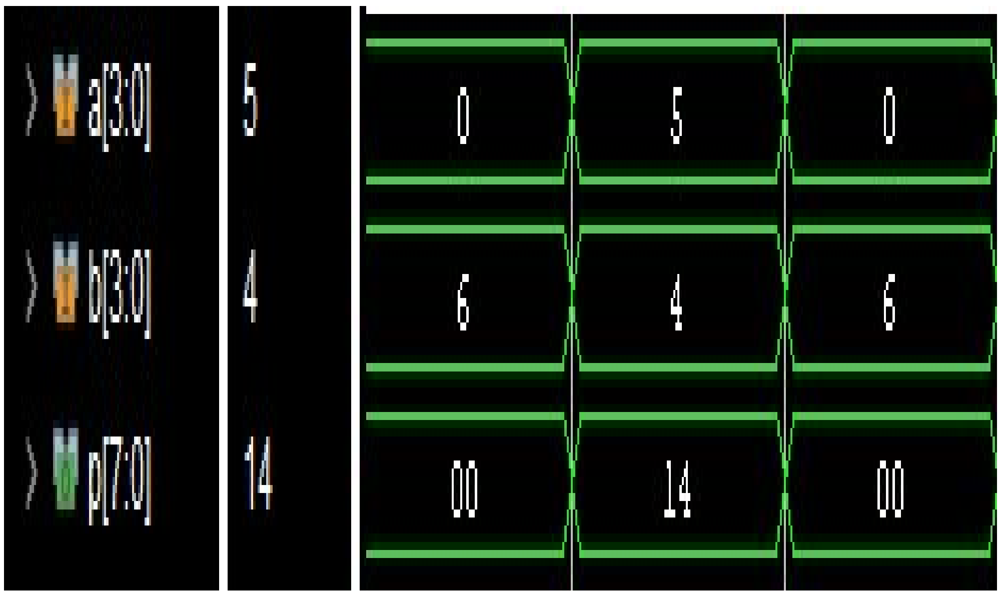


Figure 3.6: -Waveform of 4bits Array Multiplier

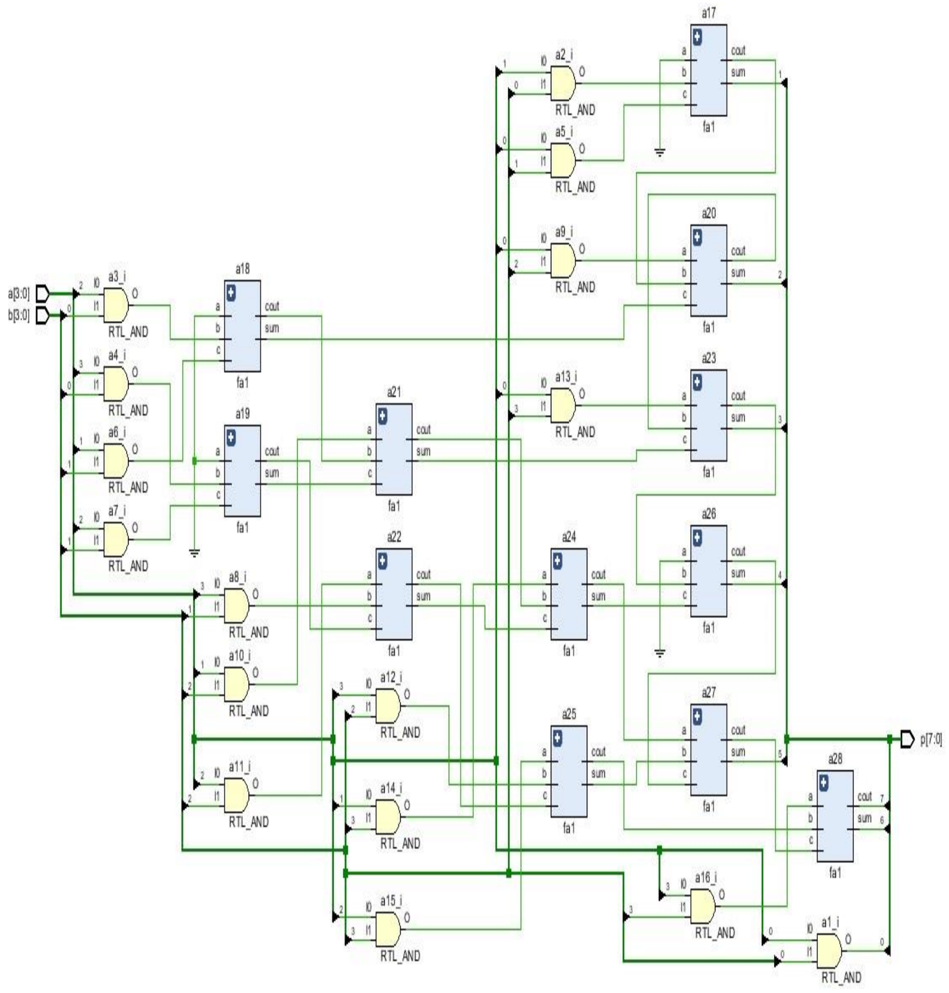


Figure 3.7: - SC of 8x8 Array Multiplier

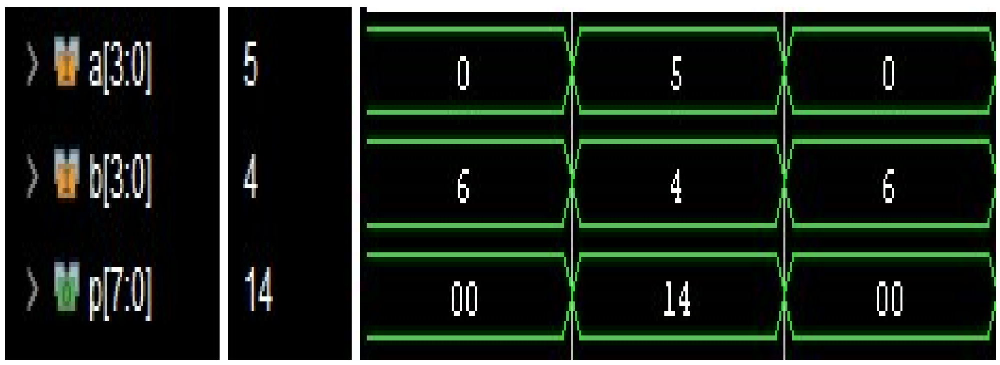


Figure 3.8: - Waveform of 8x8Array Multiplier

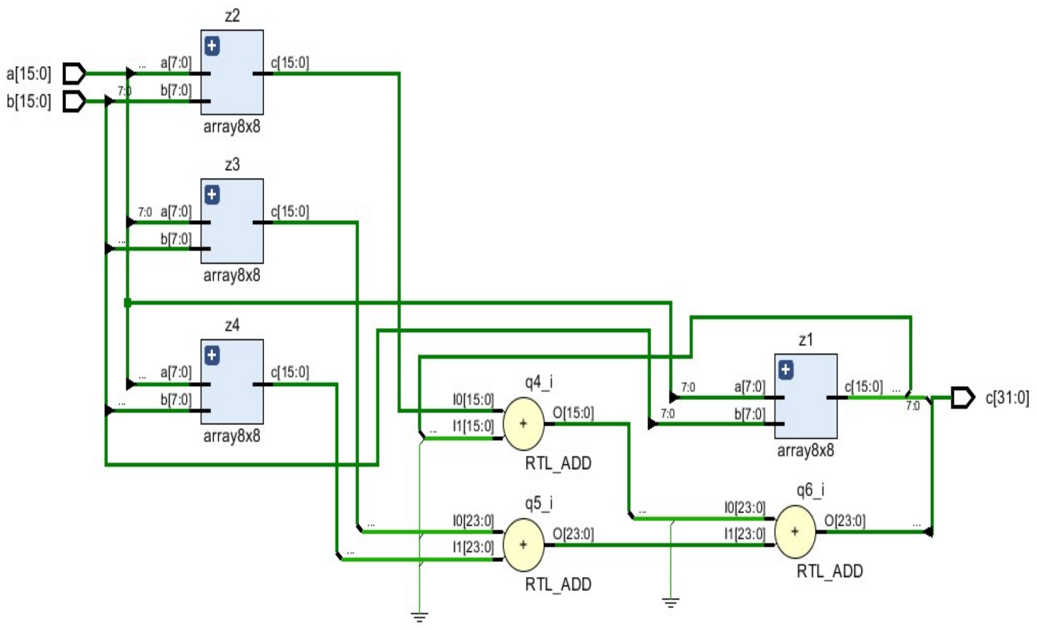
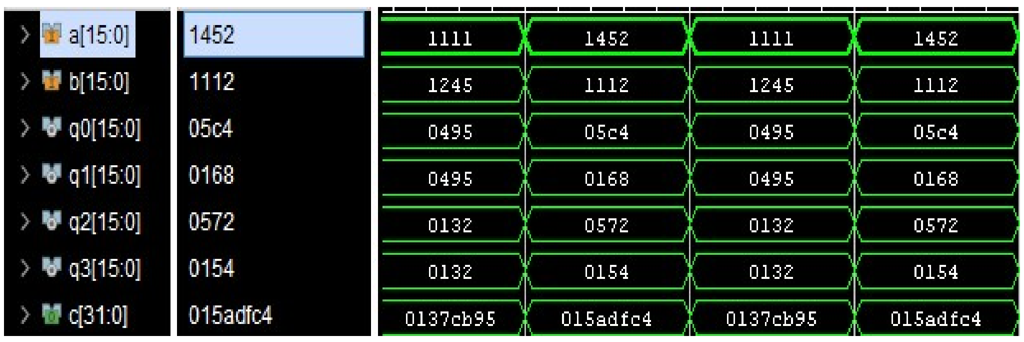


Figure 3.9 SC of 16x16 Array Multiplier



```
a[15:0]    1452      1111      1452      1111      1452
b[15:0]    1112      1245      1112      1245      1112
q0[15:0]   05c4      0495      05c4      0495      05c4
q1[15:0]   0168      0495      0168      0495      0168
q2[15:0]   0572      0132      0572      0132      0572
q3[15:0]   0154      0132      0154      0132      0154
c[31:0]    015adfc4  0137cb95  015adfc4  0137cb95  015adfc4
```

Figure 3.10 Waveform of 16x16Array Multiplier

Table 3.1: Comparative table of device utilization Array Multiplier

PARAMETERS	2x2	4x4	8x8	16x16
No. of Slices	1	5	30	57
No. of LUTs	2	15	108	193
No. of IOB	8	16	32	64
Power	0.988 W	3.947 W	14.540 W	25.811 W

The table shows the utilization of Array Multipplier for different-differnet bits.

VEDIC MULTIPLIER

The Urdhva Tiryakbhyam Sutra was utilised in this. In Urdhva Tiryakbyham Sutra, "Vertically and Crosswise" is the actual meaning. This sutra is applicable to all multiplication situations. The partial products and their total are computed simultaneously using this method.

The proposed multipliers simulated **2*2, 4*4, 8*8, 16*16** Vedic Multiplier using designed Half and Full Adder in Vivado.

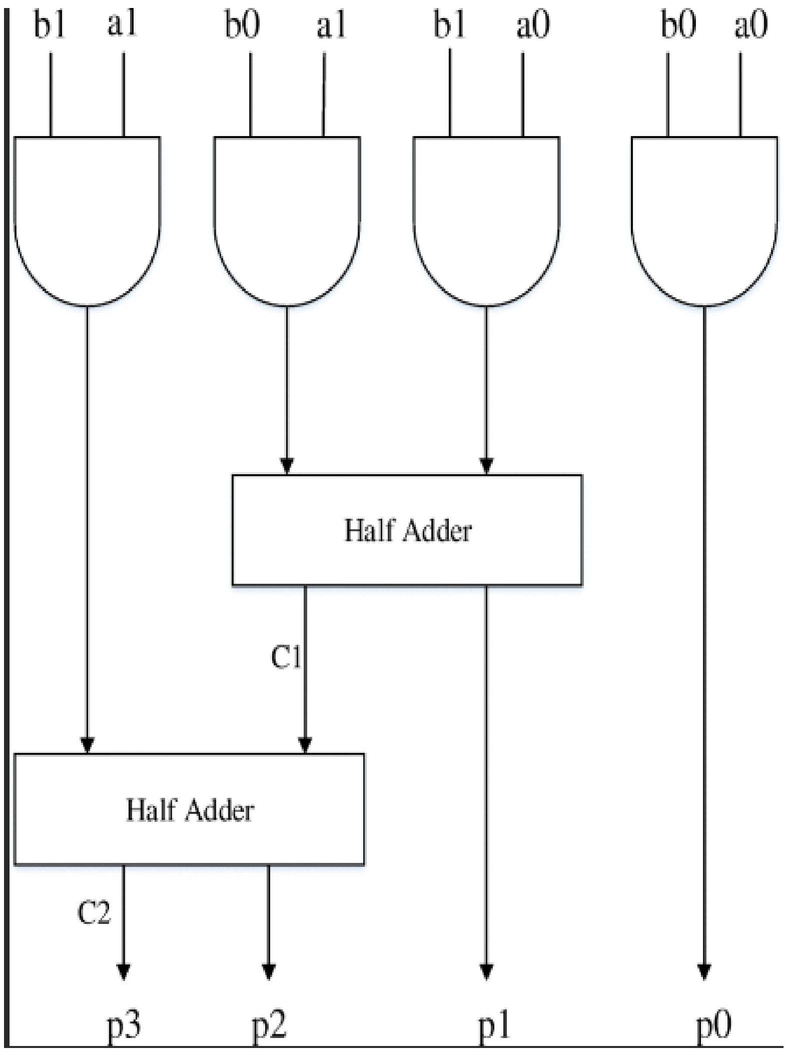


Figure 3.11: - Circuit Diagram of Vedic Multiplier

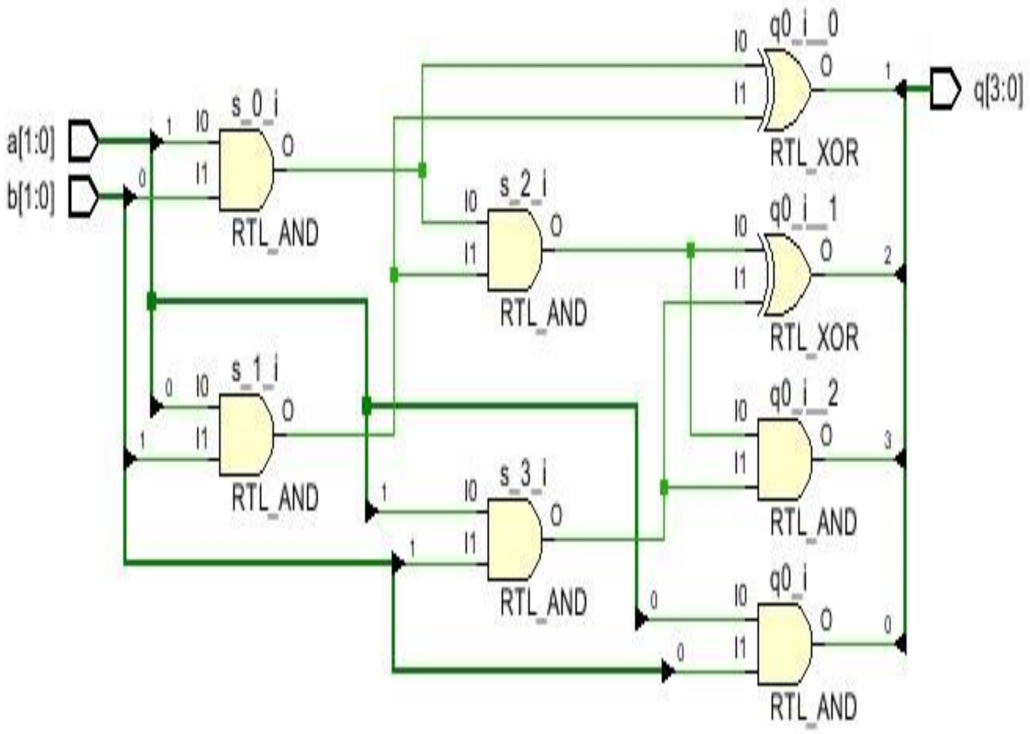


Figure 3.12: -SC of 2bits Vedic Multiplier

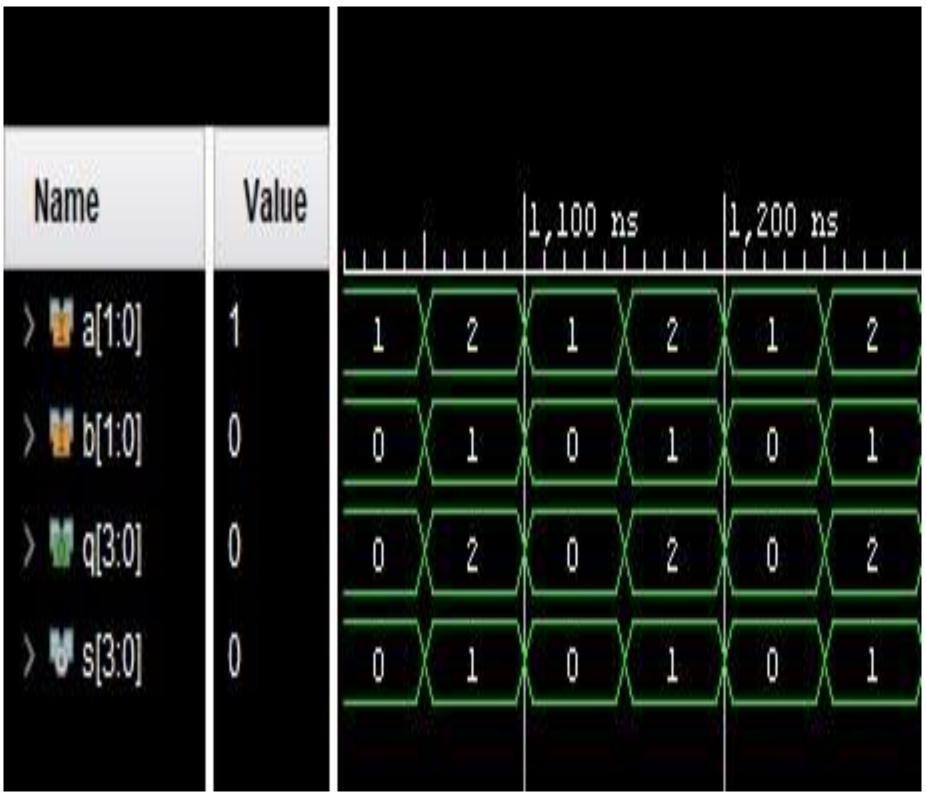


Figure 3.13: **-** Waveform of 2bits Vedic Multiplier

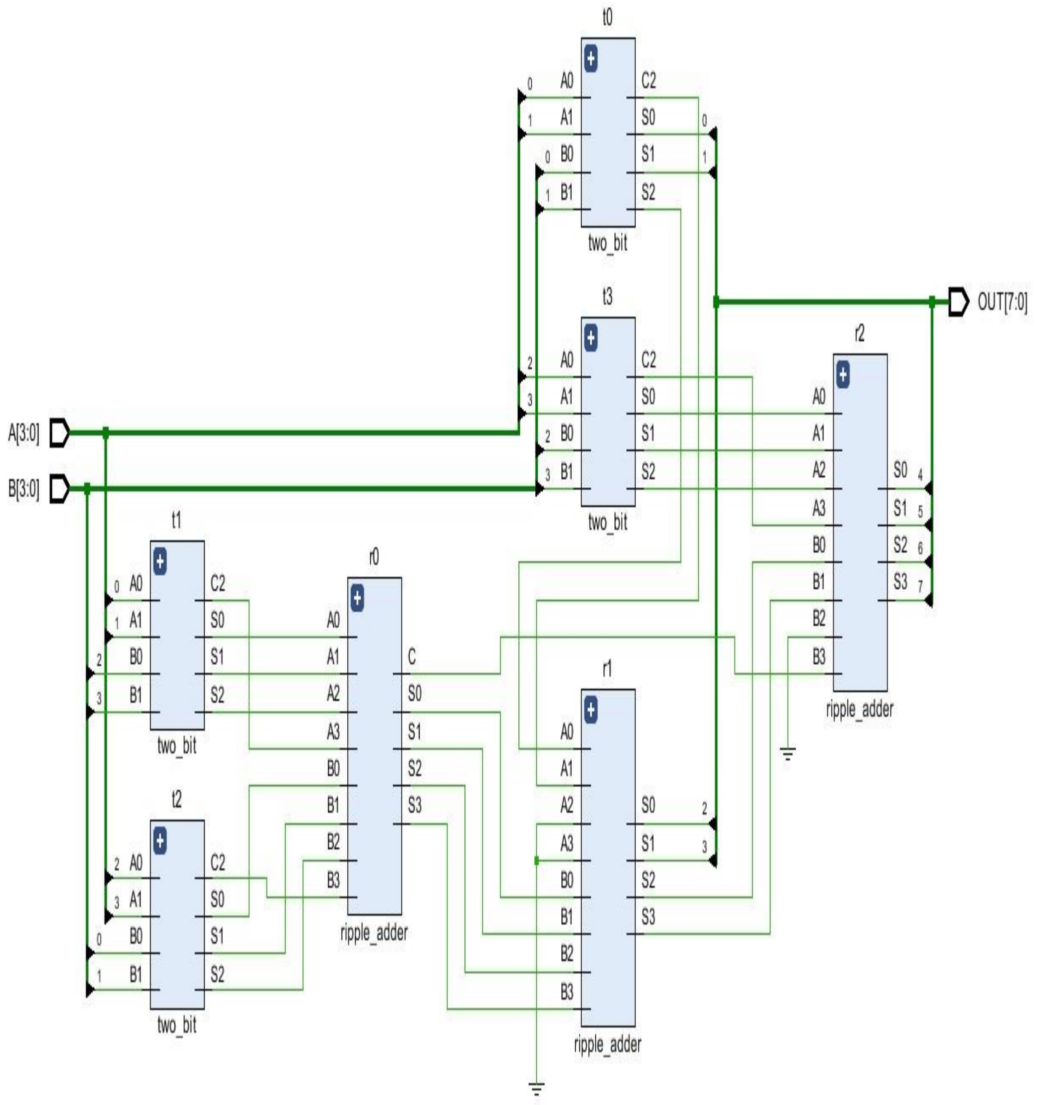
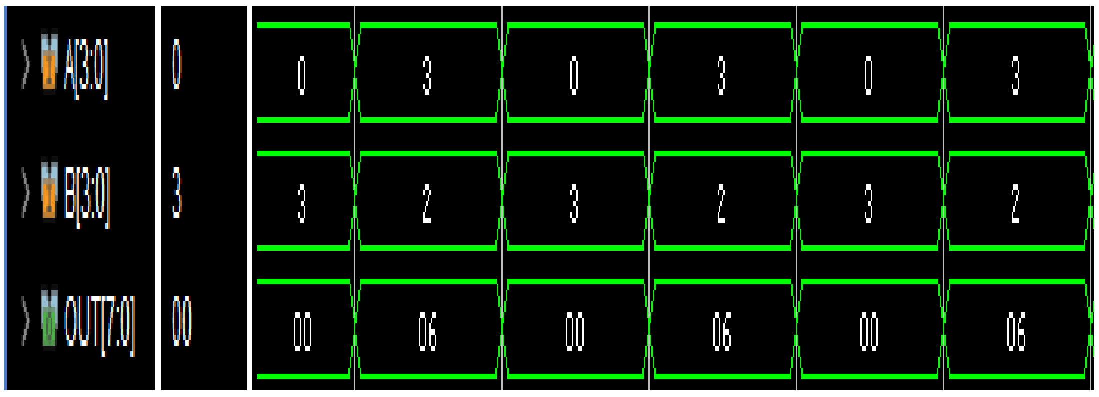


Figure 3.14: -SC of 4bits Vedic Multiplier



```
A[3:0]    0    0  3  0  3  0  3
B[3:0]    3    3  2  3  2  3  2
OUT[7:0]  00   00 06 00 06 00 06
```

Figure 3.15: - Waveform of 4bits Vedic Multiplier

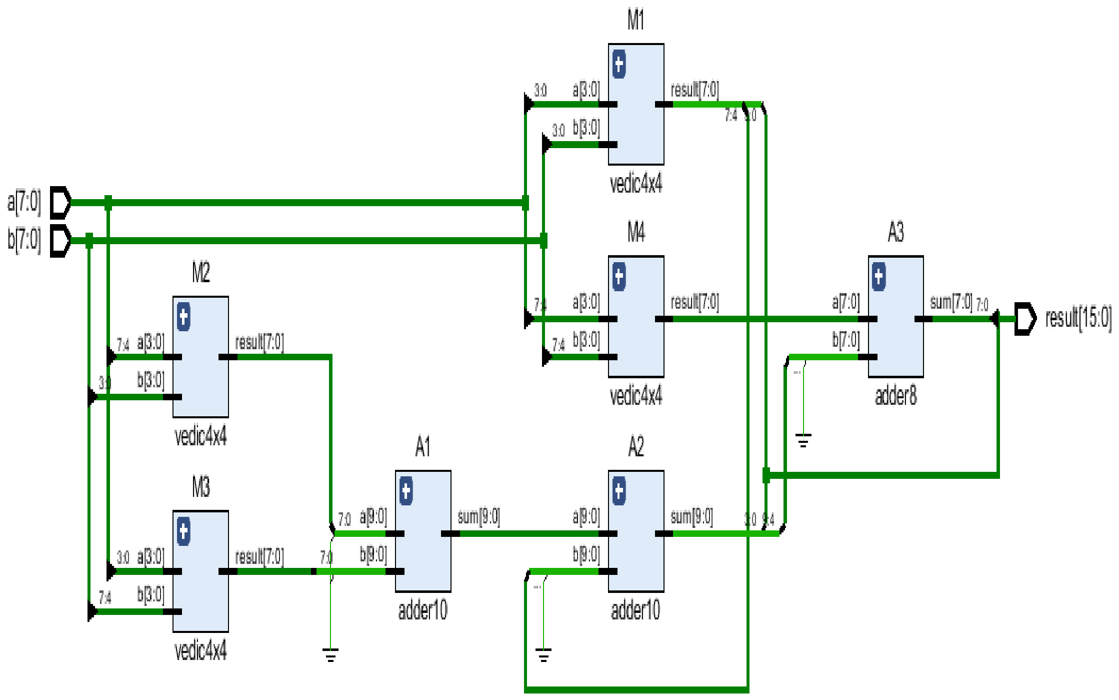


Figure 3.16: -SC of 8bits Vedic Multiplier

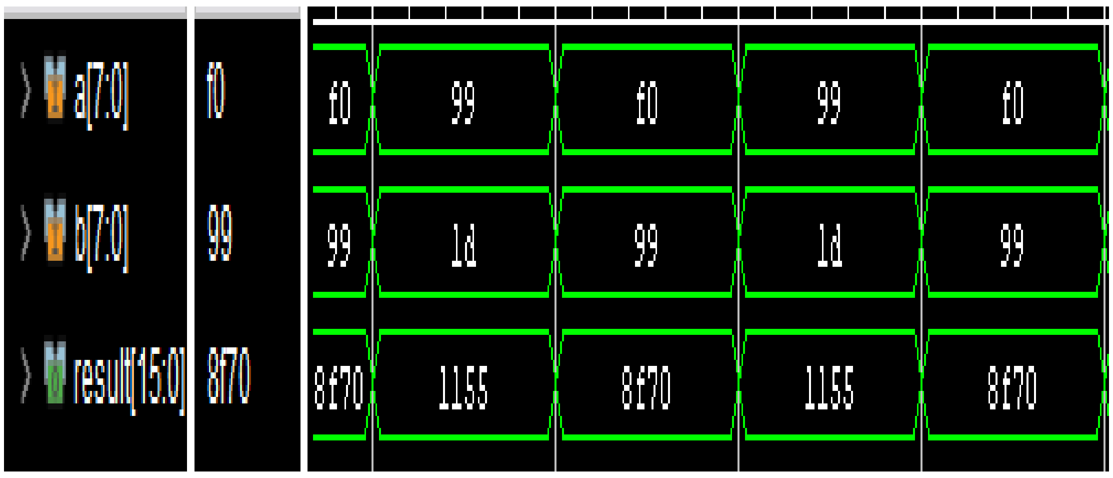


Figure 3.17: - Waveform of 8bits Vedic Multiplier

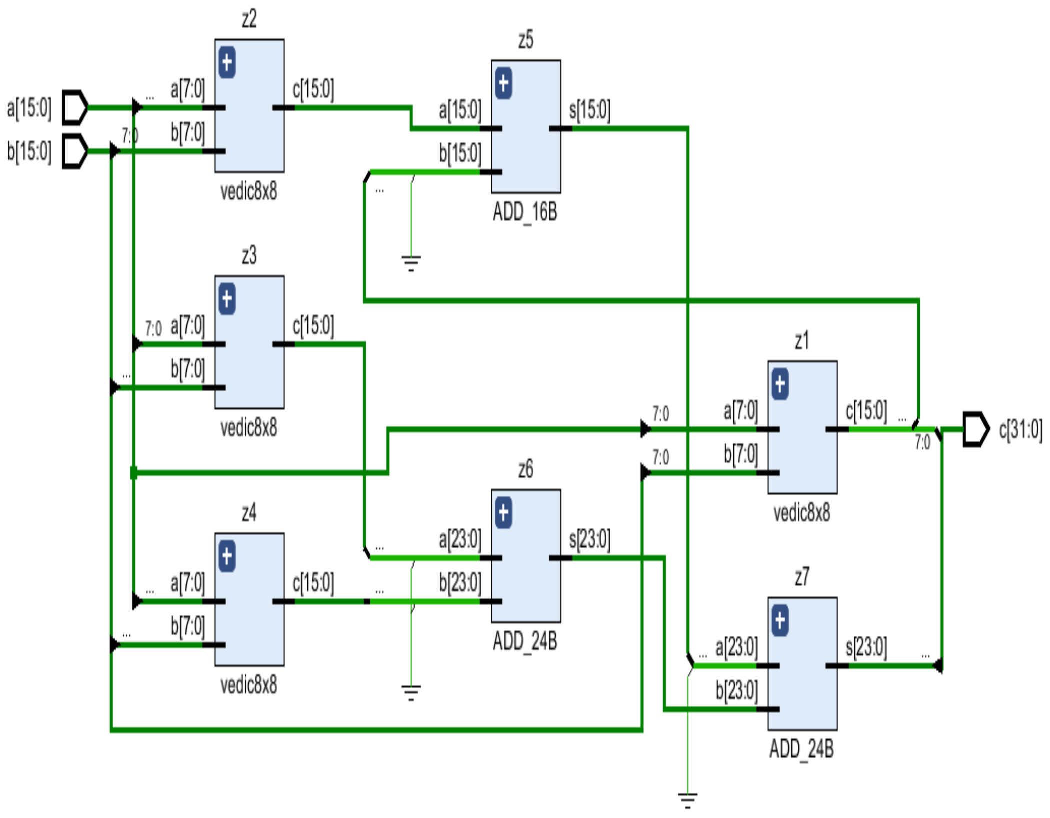
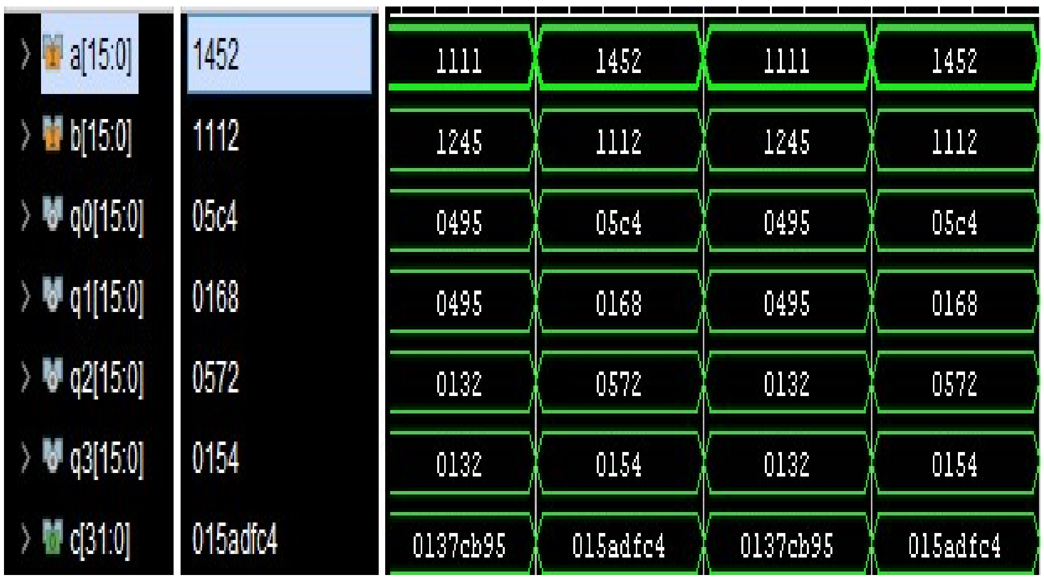


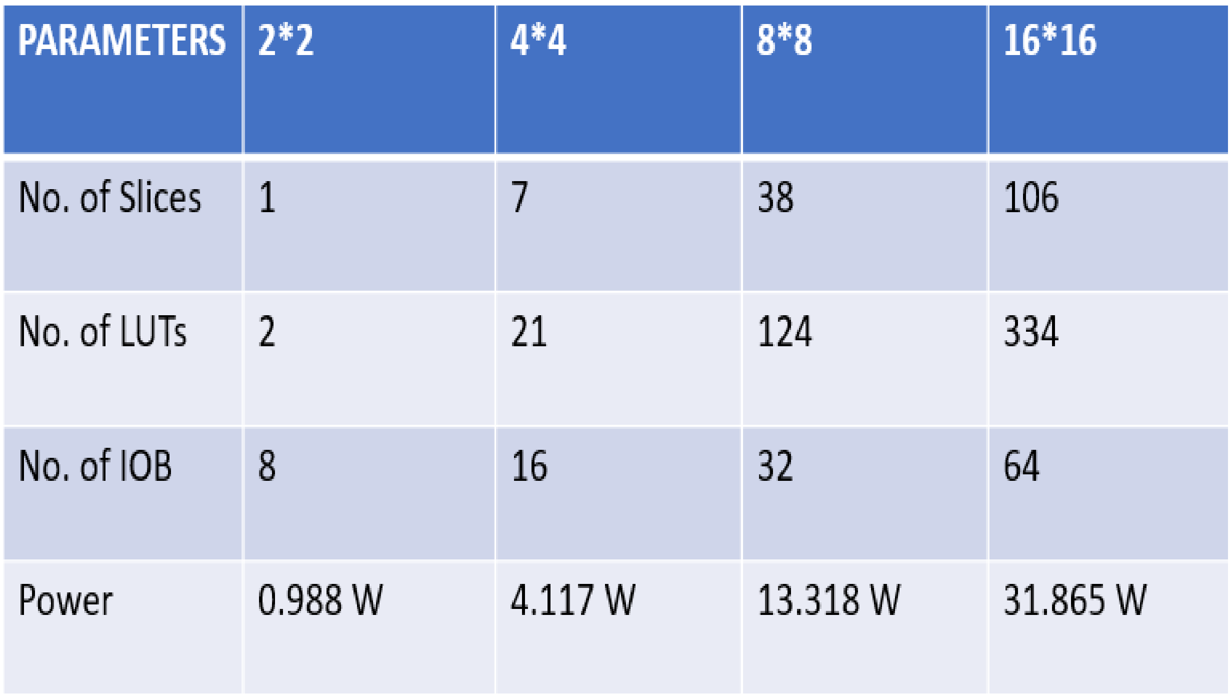
Figure 3.18: -SC of 16x16 Vedic Multiplier



a[15:0]	1452	1111	1452	1111	1452
b[15:0]	1112	1245	1112	1245	1112
q0[15:0]	05c4	0495	05c4	0495	05c4
q1[15:0]	0168	0495	0168	0495	0168
q2[15:0]	0572	0132	0572	0132	0572
q3[15:0]	0154	0132	0154	0132	0154
c[31:0]	015adfc4	0137cb95	015adfc4	0137cb95	015adfc4

Figure 3.19: - Waveform of 16x16Vedic Multiplier

Table 3.2: Comparative table of device utilization Vedic Multiplier



PARAMETERS	2*2	4*4	8*8	16*16
No. of Slices	1	7	38	106
No. of LUTs	2	21	124	334
No. of IOB	8	16	32	64
Power	0.988 W	4.117 W	13.318 W	31.865 W

The table shows the utilization of Vedic Multiplier for different-different bits.

WALLACE TREE MULTIPLIER

However, the Wallace multiplier is distinct from the array multiplier in that it adds up all partial products. The Wallace tree multiplier has multiple stages where all partial products are added.

Each level accounts for the addition of three rows in parallel, which expedites the process. Any group that has less than three rows is kept the same and moves directly on to the next addition step.

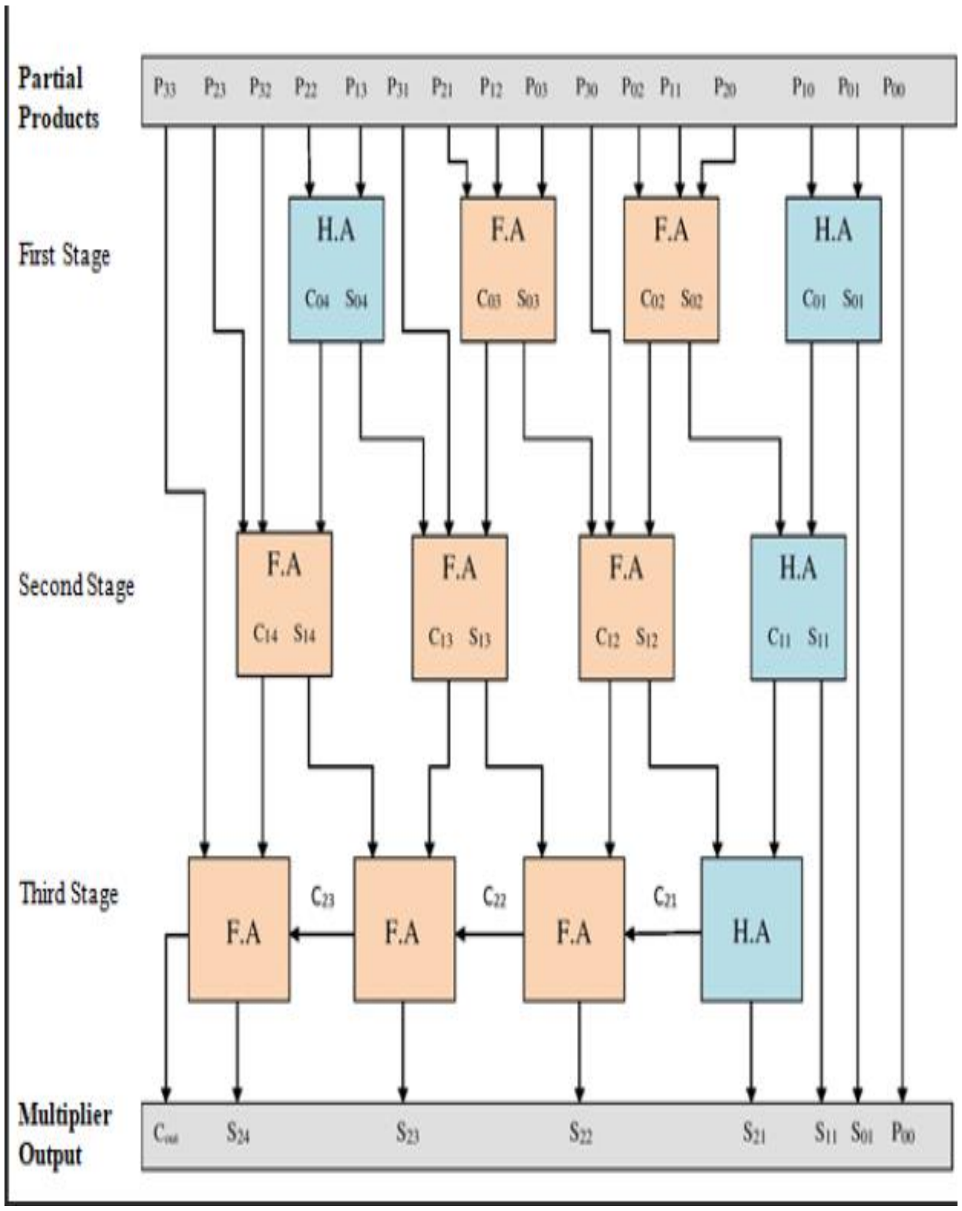


Figure 3.2: - WTM

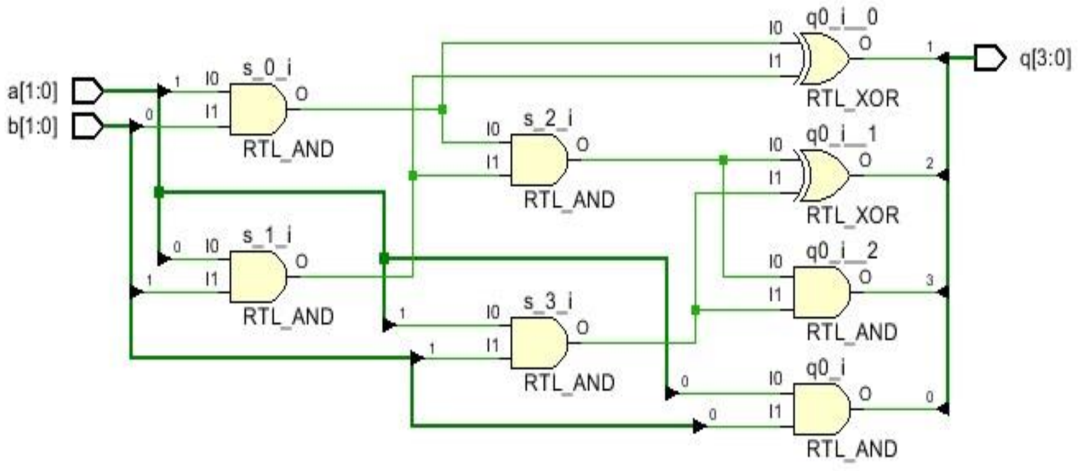
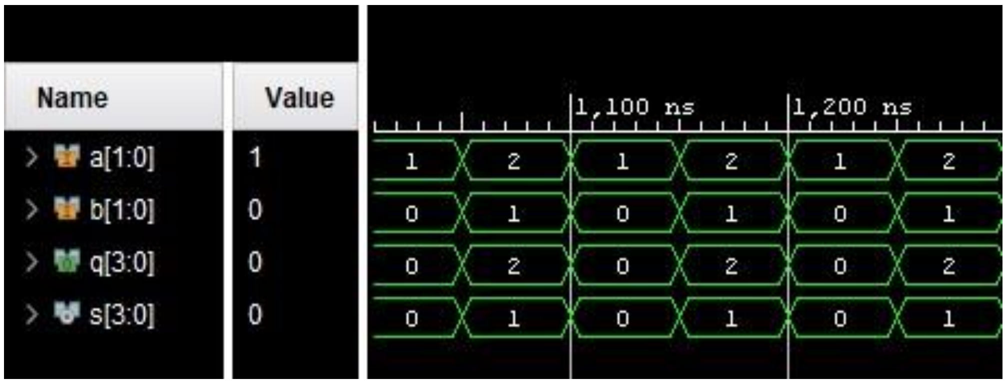


Figure 3.21: -SC of 2bits WTM



Name	Value						
a[1:0]	1	1	2	1	2	1	2
b[1:0]	0	0	1	0	1	0	1
q[3:0]	0	0	2	0	2	0	2
s[3:0]	0	0	1	0	1	0	1

Figure 3.22: -Waveform of 2bits WTM

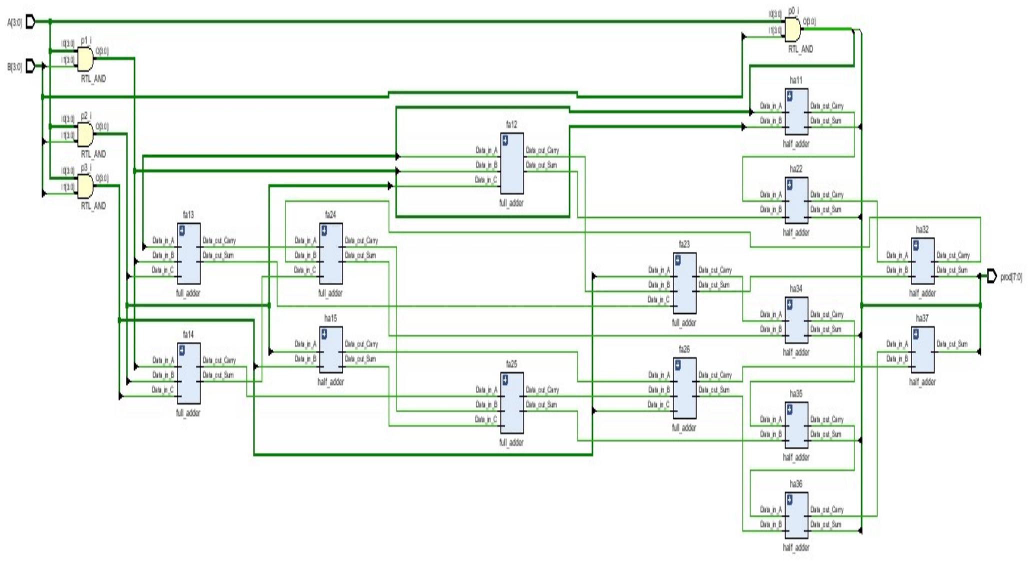


Figure 3.23: -SC of 4bits WTM

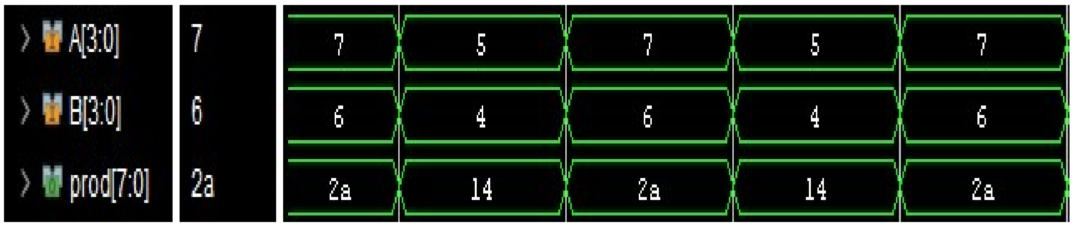


Figure 3.24: - Waveform of 4bits WTM

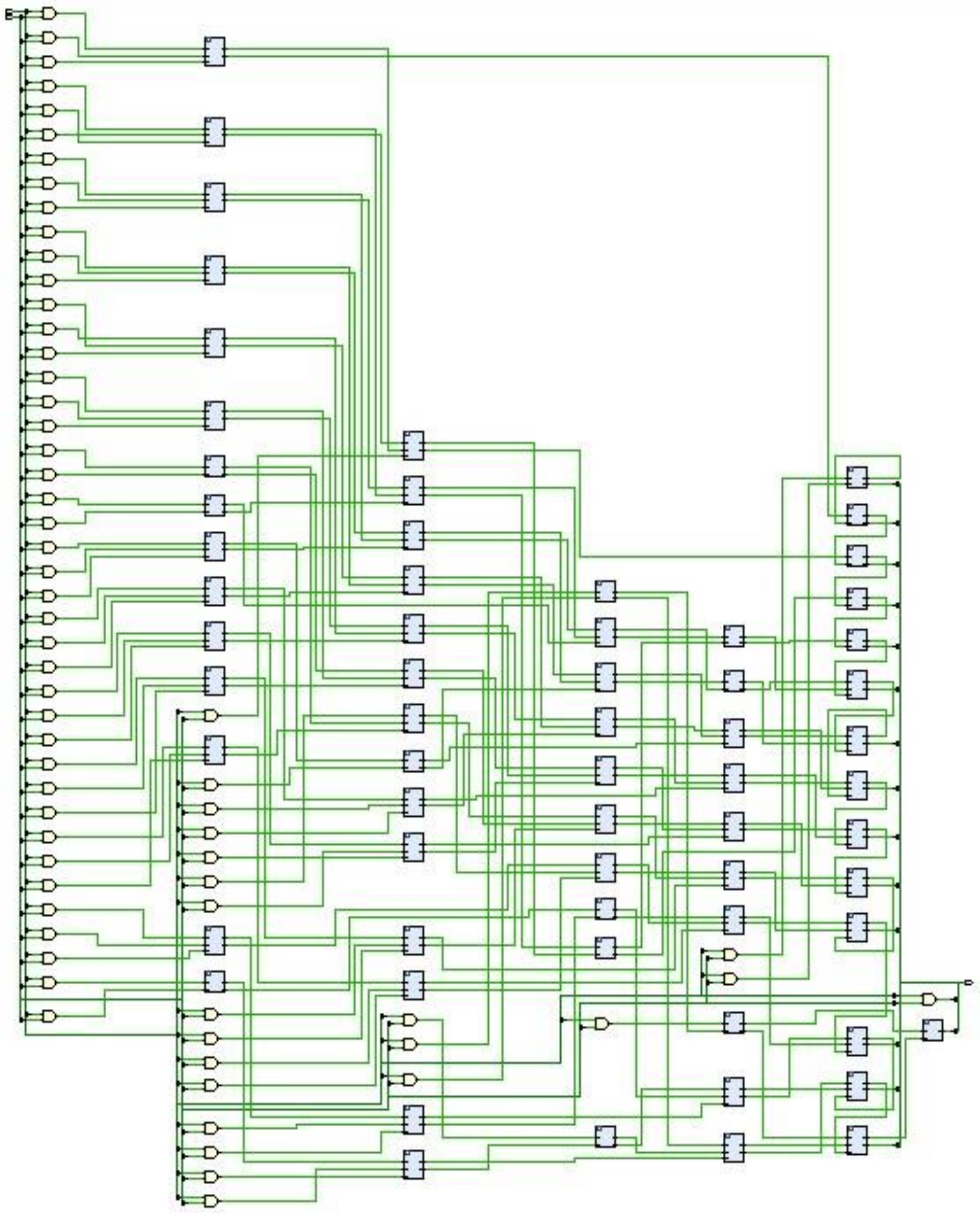


Figure 3.25: - SC of 8bits WTM

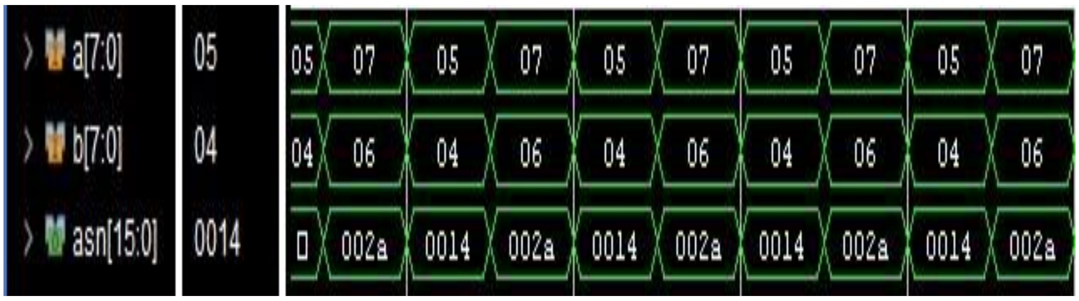


Figure 3.26: -Waveform of 8bits WTM

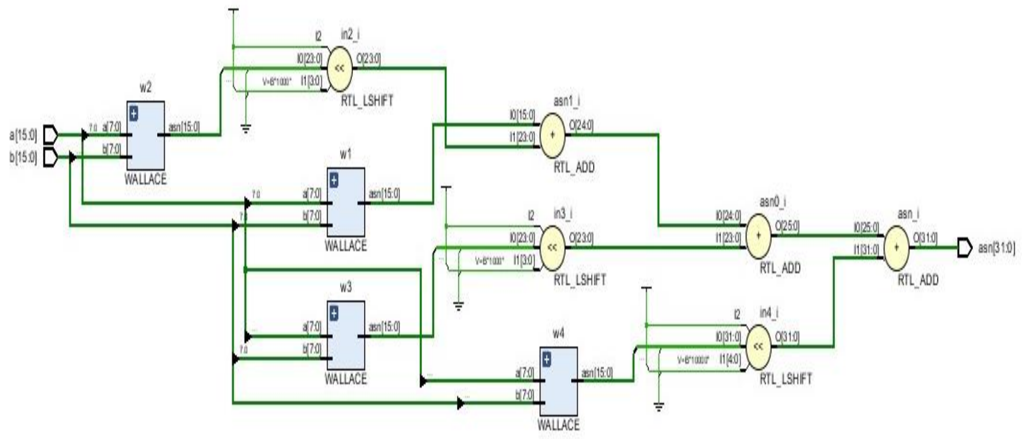


Figure 3.27: - SC of 16bits WTM

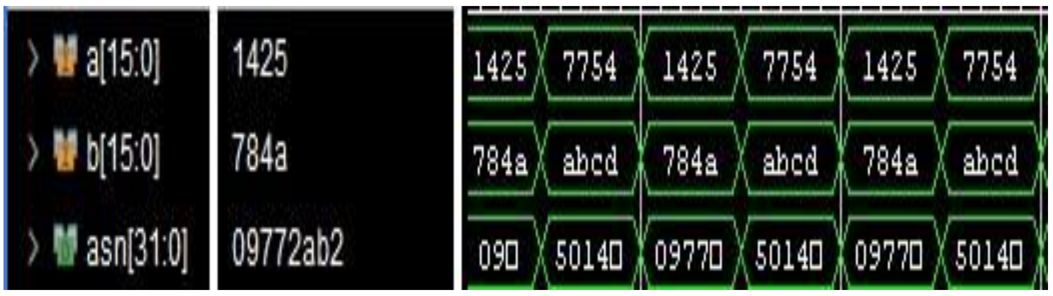


Figure 3.28: -Waveform of 16x16 WTM

Table 3.3: Comparative table of Utilization of Wallace Tree Multiplier

PARAMETERS	2x2	4x4	8x8	16x16
No. of Slices	1	5	24	100
No. of LUTs	2	17	88	346
No. of IOB	8	16	32	64
Power	0.988 W	4.099 W	13.220 W	31.153 W

This table shows the utilization of WTM for different-different bits

In this study, we designed Three different Multipliers —array, Vedic, and Wallace tree with varying bit sizes of 2x2, 4x4, 8x8, and 16x16. A comprehensive comparison based on specific parameters (No. of slices, No. of LUTs, Power) was conducted, leading to the determination that the Wallace tree multiplier outperformed the alternative.

CHAPTER 4

IMPLEMENTATION OF WALLACE TREE MULTIPLIER USING DIFFERENT LOGIC.

Further analysis was performed on the Wallace tree multiplier, considering four distinct gate types—**AOI, NAND, NOR, and OAI.**

- **Implementation WTM using AOI Logic: -**

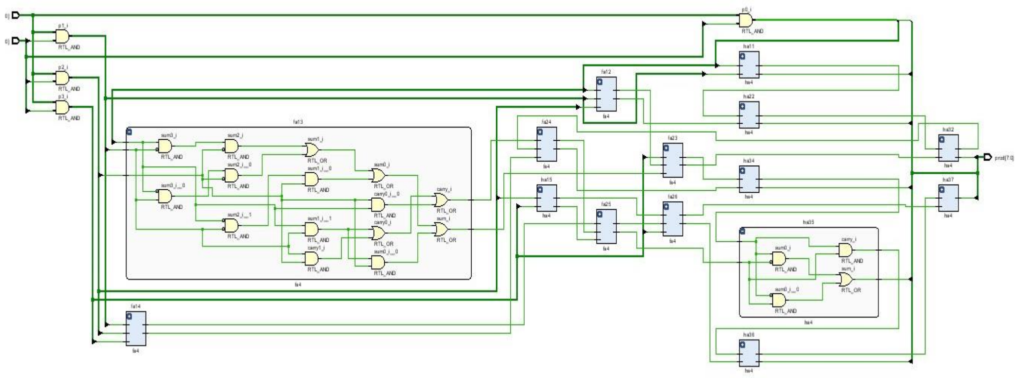


Figure 4.1: -SC of 4X4 WTM AOI Logic**.**

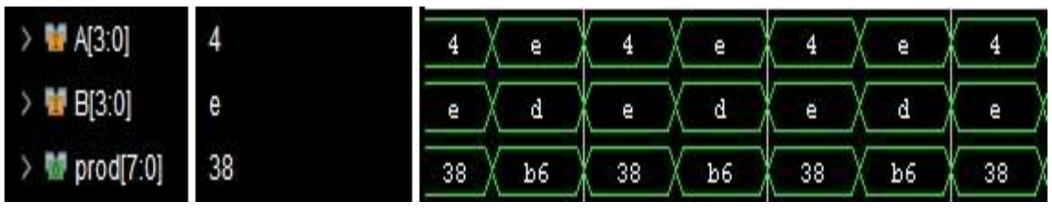


Figure 4.2: - Waveform of 4x4Wallace Tree Multiplier using AOI Logic.

- **Implementation of WTM using NAND Logic: -**

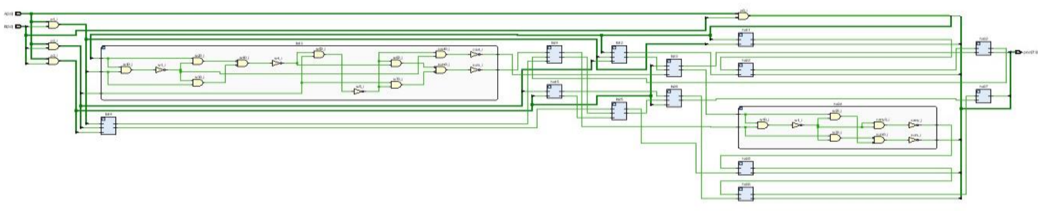


Figure 4.3: -SC of 4bits WTM using NAND Logic.

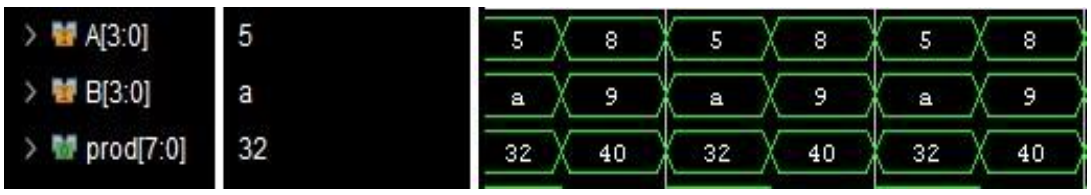


Figure 4.4: -Waveform of 4bits WTM using NAND Logic.

- ## Implementation of WTM using NOR Logic: -

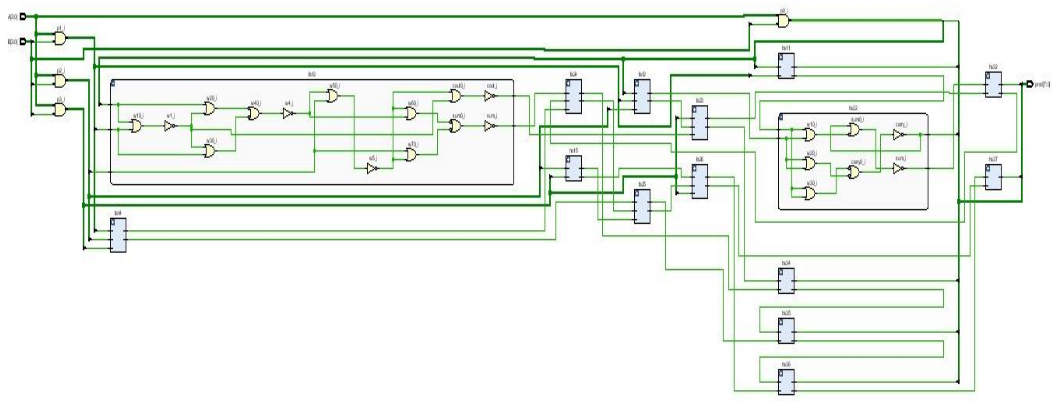
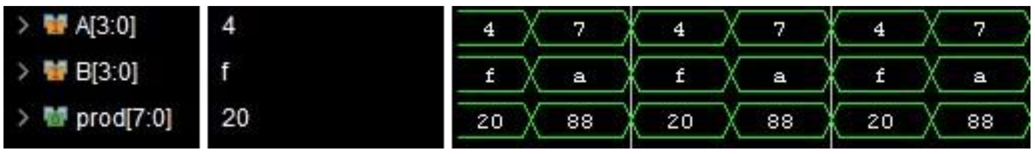


Figure 4.5: -SC of 4bits WTM using NOR Logic.



```
A[3:0]     4     4   7   4   7   4   7
B[3:0]     f     f   a   f   a   f   a
prod[7:0]  20    20  88  20  88  20  88
```

Figure 4.6: -Waveform of 4bits WTM using NOR Logic

- ## Implementation of WTM using OAI Logic: -

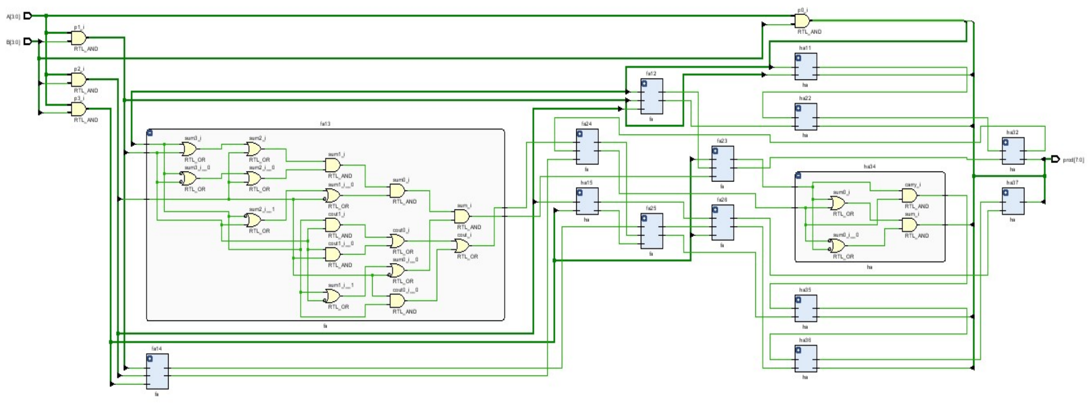


Figure 4.7: -SC of 4bits WTM using OAI Logic.

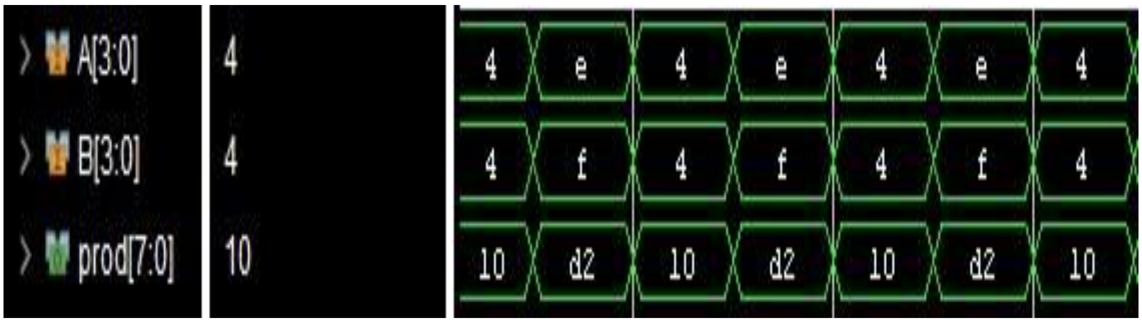


Figure 4.8: -Waveform of 4bits WTM using OAI Logic.

Through meticulous evaluation, it was established that the **NOR** gate configuration yielded the most optimal performance for the Wallace tree multiplier.

Table 4.1: Comparative table of Utilization of WTM using Different Logics.

PARAMETERS	AOI	NAND	NOR	OAI
No. of Slices	6	5	6	6
No. of LUTs	21	20	22	21
No. of IOB	16	16	16	16
Power	3.684 W	3.419 W	2.256 W	3.684 W

This table shows that Wallace Tree using NOR Logic consumes the minimum power than the other logics.

In this study, we designed WTM Using NOR, NAND, AOI and OAI Logic for 4-bit. A comprehensive comparison based on specific parameters (No. of slices, No. of LUTs, Power) was conducted, leading to the determination that the WTM using NOR logic outperformed the alternative.

IMPLEMENTATION OF WTM USING NOR GATE ON FPGA

It was established that the **NOR** gate configuration yielded the most optimal performance for the Wallace tree multiplier and final generated bit stream will be used to burn FPGA.

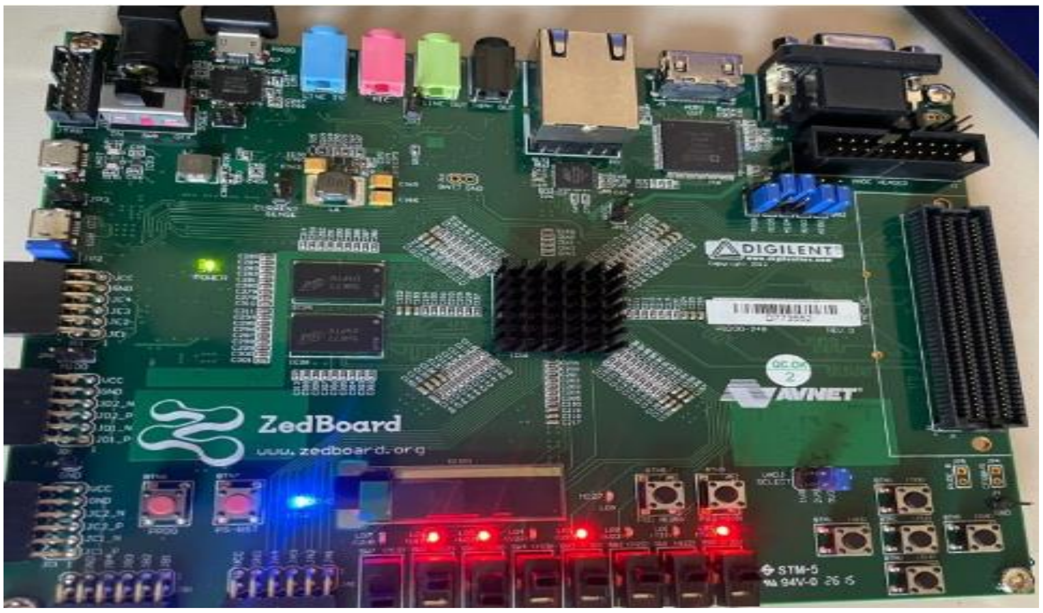


Figure 4.9: -: FPGA Implementation of 4bits Wallace Tree Multiplier using NOR Logic.

We took the input of the multiplier as 1111 * 1010 and the output for the same is 10010110.

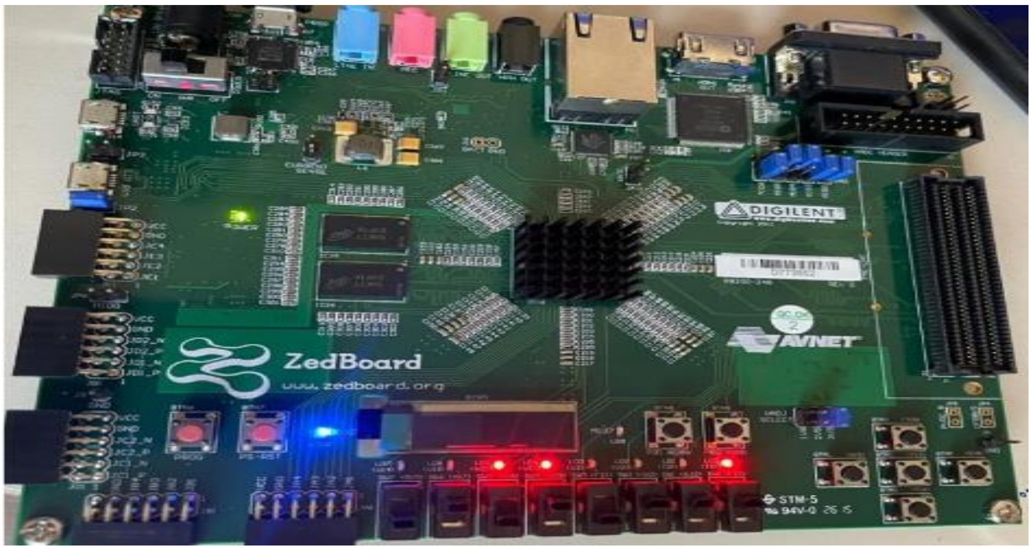


Figure 4.10: FPGA Implementation of 4bits Wallace Tree Multiplier using NOR Logic.

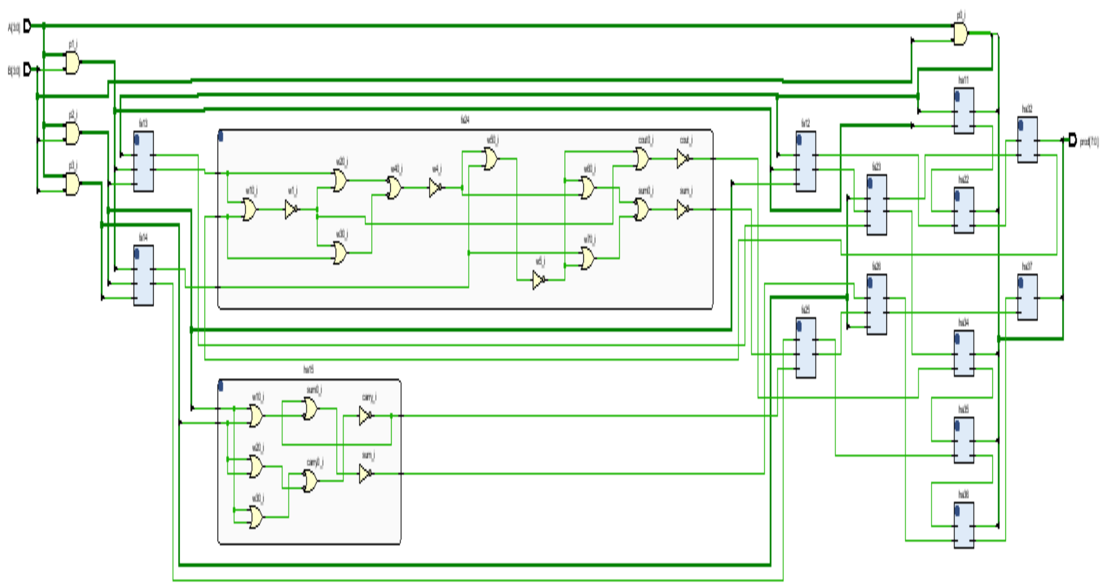
We took the input of the multiplier as 1110 * 1010 and the output for the same is 10001100.

CHAPTER -5

IMPLEMENTATION OF WALLACE TREE MULTIPLIER USING NOR LOGIC.

Further analysis was performed on the Wallace tree multiplier, considering NOR gate type— **4-bit,8-bit,16-bit and 32-bit**

- **Implementation ofWTM using NOR Logic for 4-bit:**



.

Figure 5.1: - SC of 4bits WTM using NOR Logic

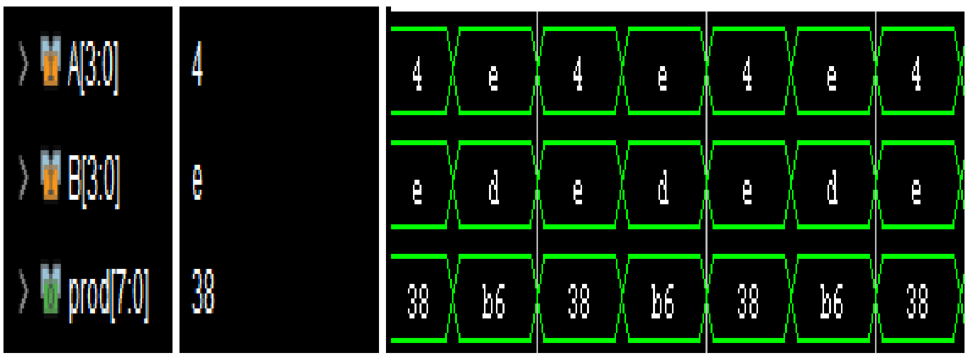


Figure 5.2: - Waveform of 4x4 WTM using NOR Logic.

- **Implementation of WTM using NOR Logic for 8-bit: -**

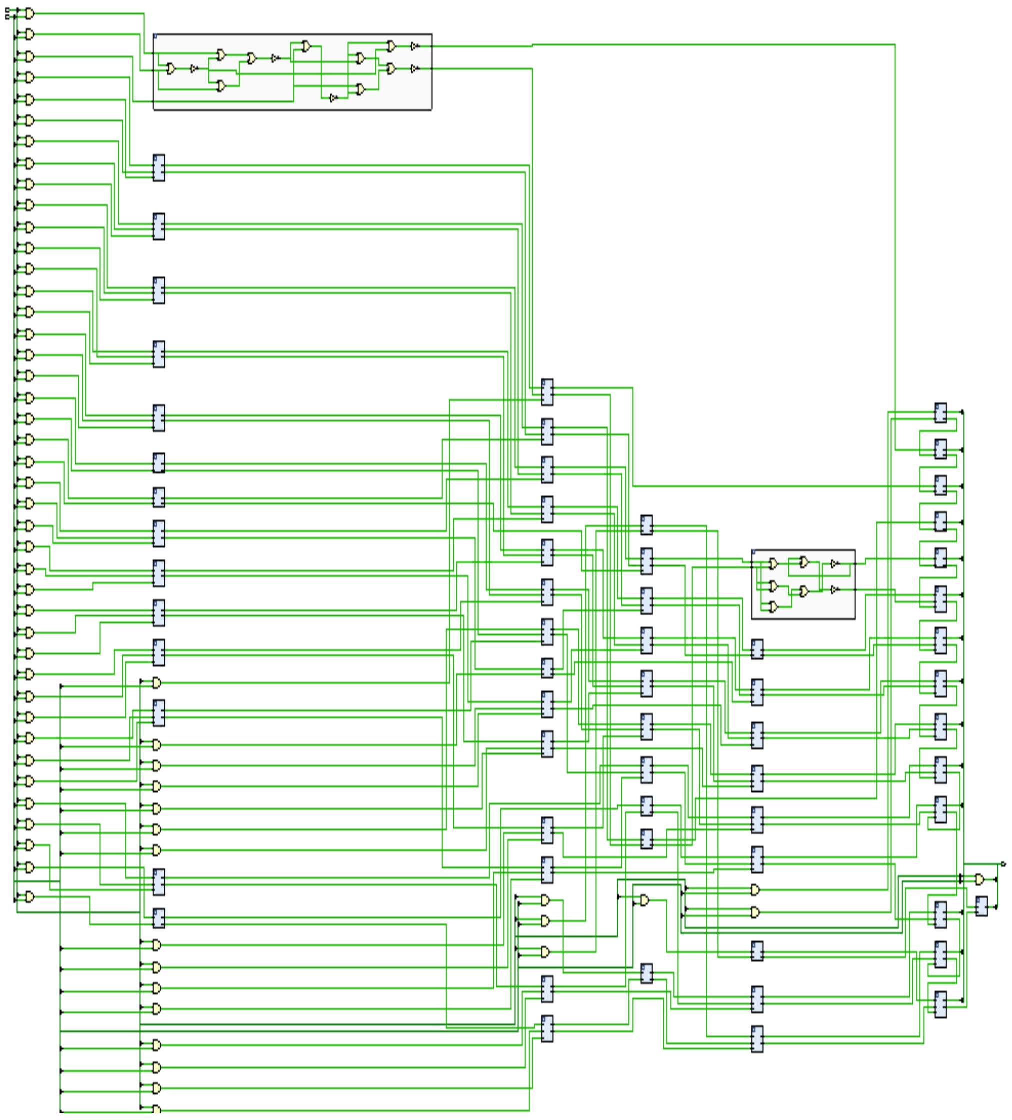
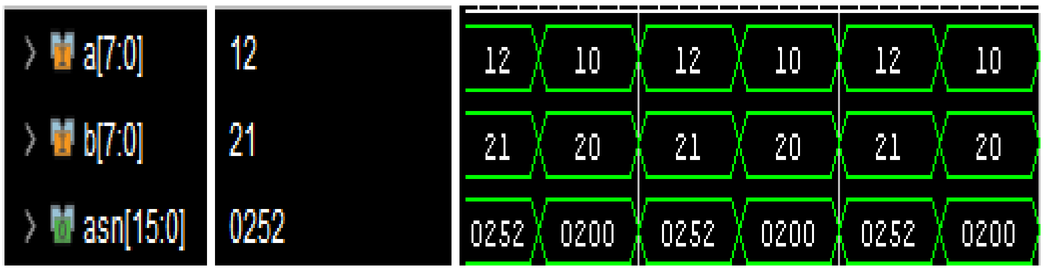


Figure 5.3: - SC of 8bits WTM using NOR Logic



```
a[7:0]      12     12    10    12    10    12    10
b[7:0]      21     21    20    21    20    21    20
asn[15:0]   0252   0252  0200  0252  0200  0252  0200
```

Figure 5.4: - Waveform of 8bits WTM using NOR Logic.

- **Implementation of WTM using NOR Logic for 16-bit: -**

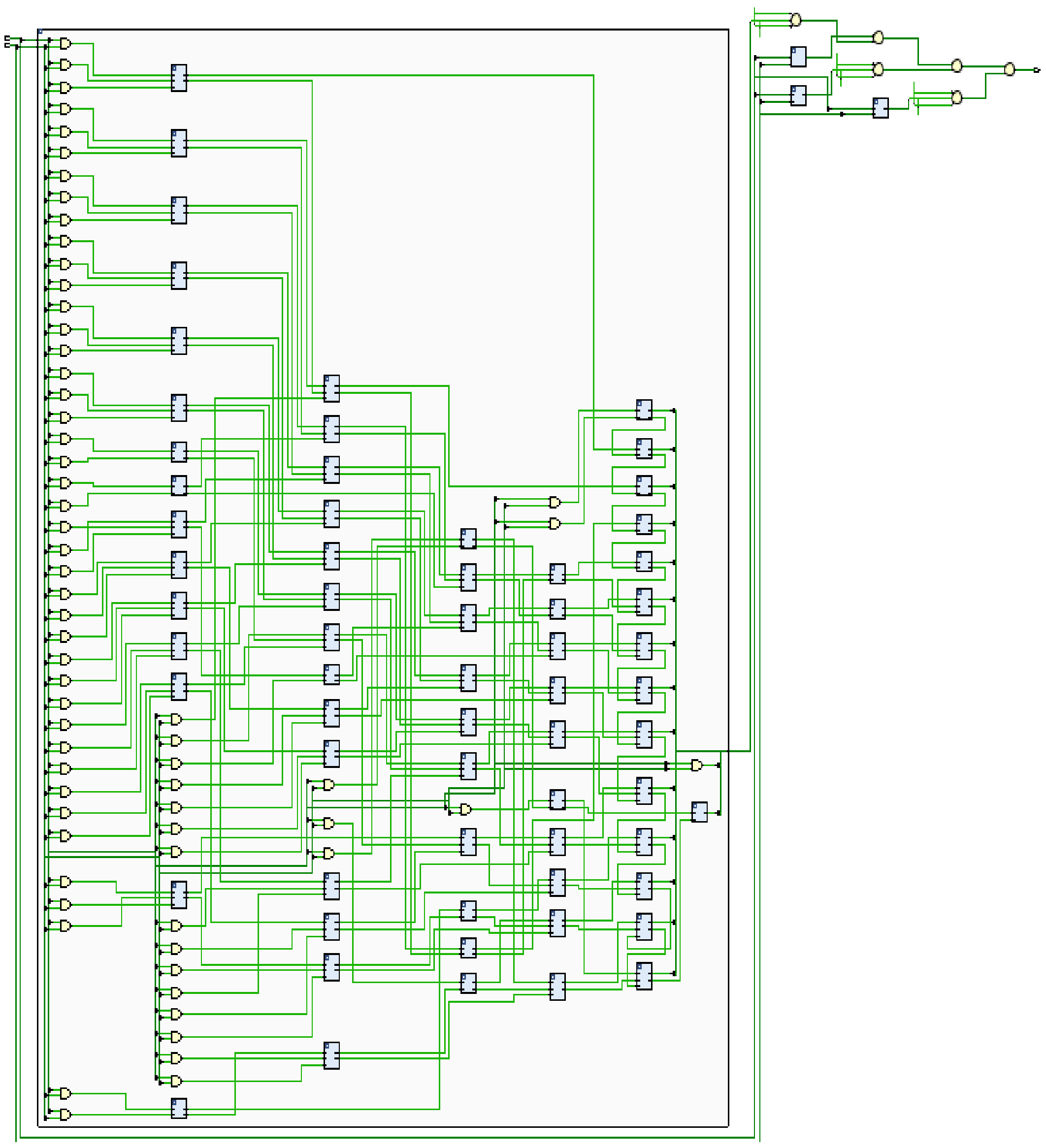


Figure 5.5: - SC of 16-bits WTM using NOR Logic

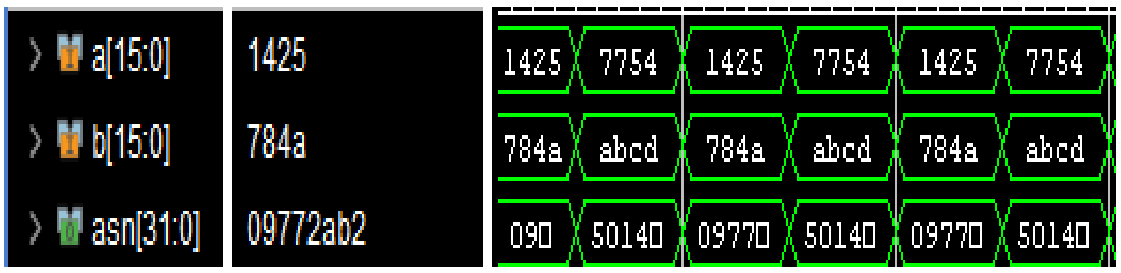


Figure 5.6: - Waveform of 16-bits WTM using NOR Logic.

- **Implementation of WTM using NOR Logic for 32-bit: -**

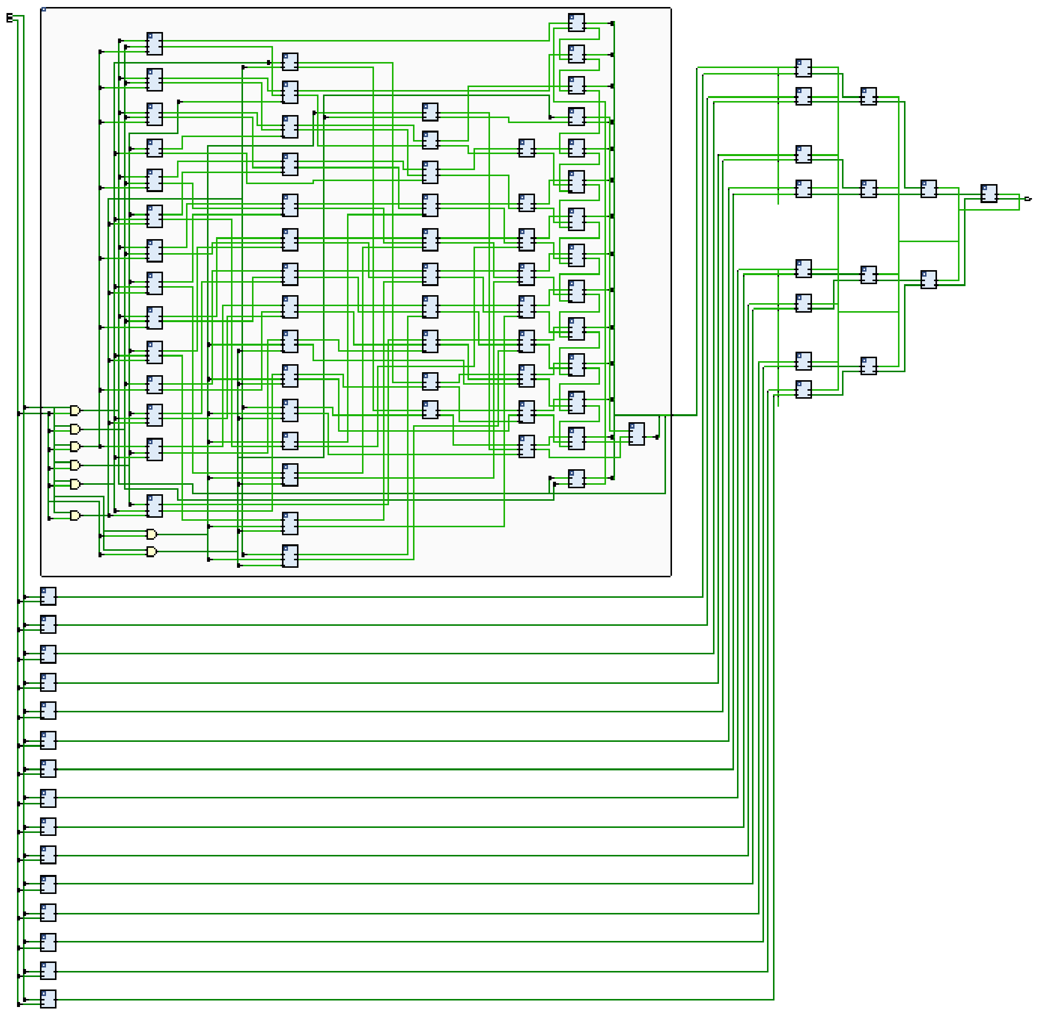


Figure 5.7: - SC of 32-bits WTM using NOR Logic

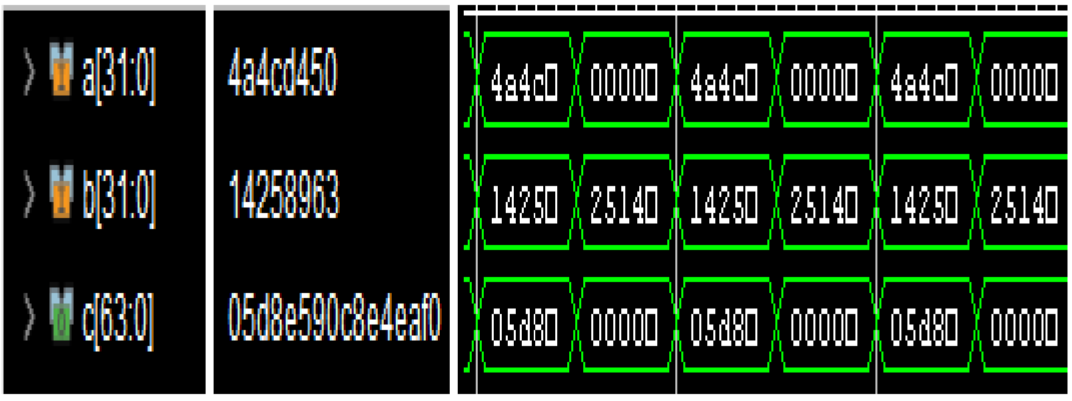
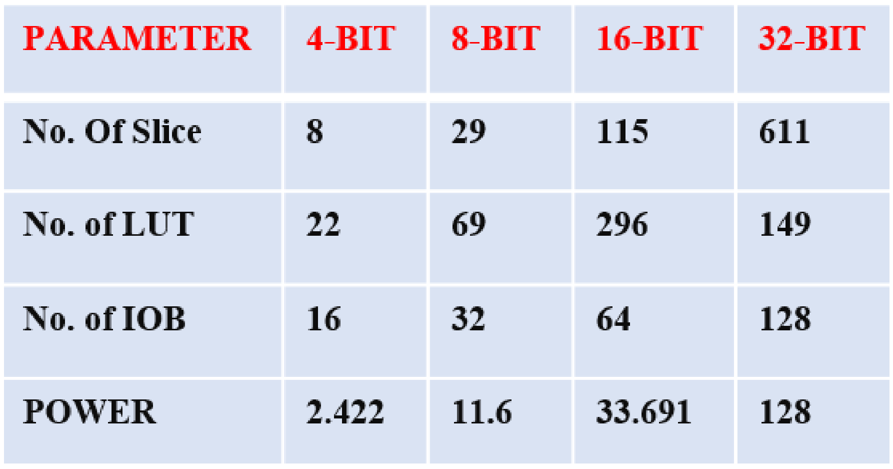


Figure 5.8: - Waveform of 32-bits WTM using NOR Logic.

Table 5.1: Comparative table of Utilization of WTM using NOR Logics for different bits.



PARAMETER	4-BIT	8-BIT	16-BIT	32-BIT
No. Of Slice	8	29	115	611
No. of LUT	22	69	296	149
No. of IOB	16	32	64	128
POWER	2.422	11.6	33.691	128

The table shows the utilization of WTM using NOR Logic for different-different bits.

In this study, we designed WTM using NOR logic varying bit sizes of 2x2, 4x4, 8x8, and 16x16. A comprehensive comparison based on specific parameters (No. of slices, No. of LUTs, Power) was conducted.

CHAPTER -6

WALLACE TREE MULTIPLIER USING 4:2 COMPRESSORS

Because digital circuit complexity is increasing, power density and heat dissipation are rising in today's (VLSI) world. The leakage current rises when the power density goes high which causes a decline reliability and lifespan of an IC. Different computing aids are used to conserve power and also help to enhance the speed of the circuit. Applications such as image processing, and data mining where these multipliers are commonly used.

a) Half-Adder

Digital circuits are fundamentally constructed using half-adders. They are also used in the process of updating arithmetic components as an optimization circuit. The precise and accurate circuit diagram representation of the Half-adder is displayed in Figure 6.1. Equations (6.1) and (6.2) are used to compute the final **sum** and **carry** for the exact/precise Half-Adder, respectively.

$$\textbf{sum} = (x1 \oplus x2) \qquad (6.1)$$

$$\textbf{carry} = (x1.x2) \qquad (6.2)$$

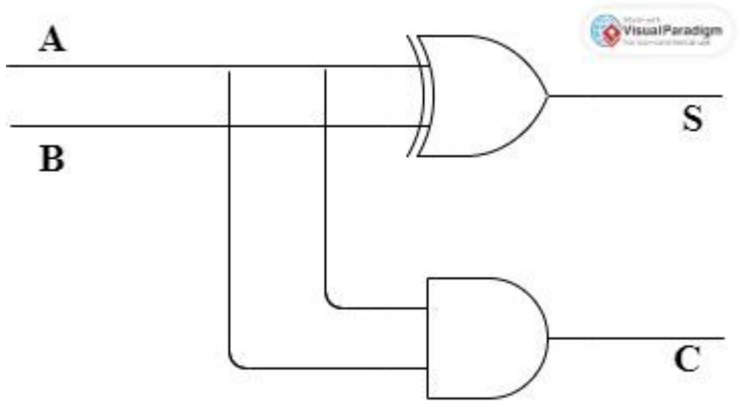


Figure 6.1: - Diagram of Half-Adder**.**

Table 6.1: Truth Table of Half-Adder

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0

1	1	0	1

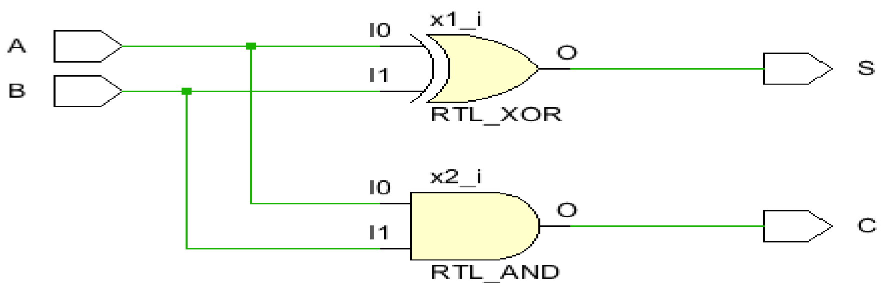


Figure 6.2: - SC of Half-Adder**.**

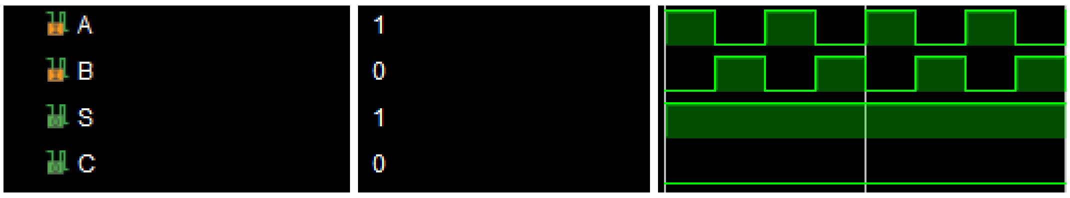


Figure 6.3: - Waveform of Half-Adder

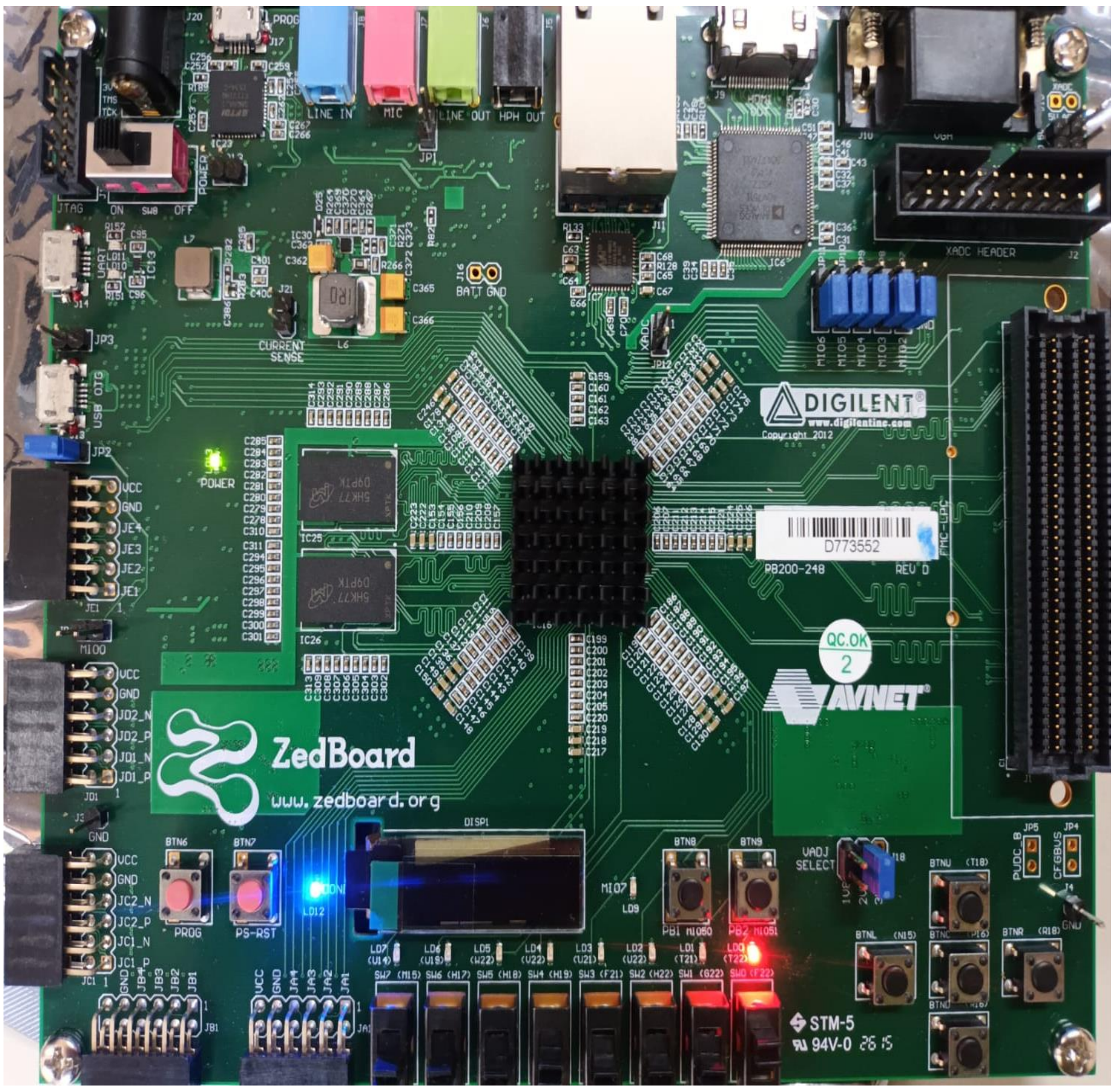


Figure 6.4: - FPGA implementation of Half-Adder**.**

b) Full-Adder

The most basic compressor is the full-adder, sometimes referred to as the 3:2 compressor. It converts three inputs (A, B, Cin) into two outputs (sum, carry), as shown in Figure 6.5. The 3:2 compressor's final **Sum** and **Carry** are determined by applying Equations (6.3) and (6.4), respectively.

$$\textbf{Sum} = (A \oplus B \oplus Cin) \qquad (6.3)$$

$$\textbf{Carry} = (A.\ B) + (B.\ Cin) + (Cin.\ A) \qquad (6.4)$$

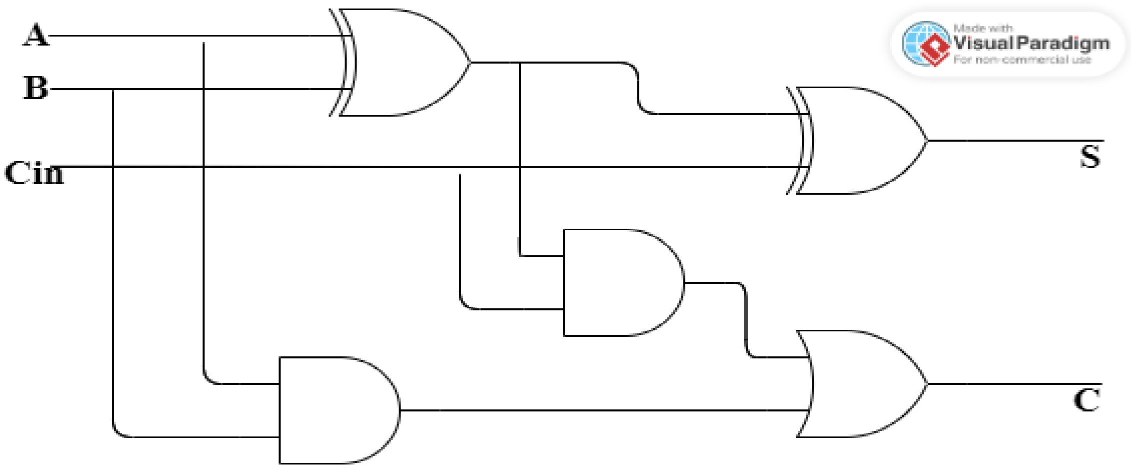


Figure 6.5: - Diagram of Full-Adder

Table 6.2: Truth Table of Full-Adder

A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

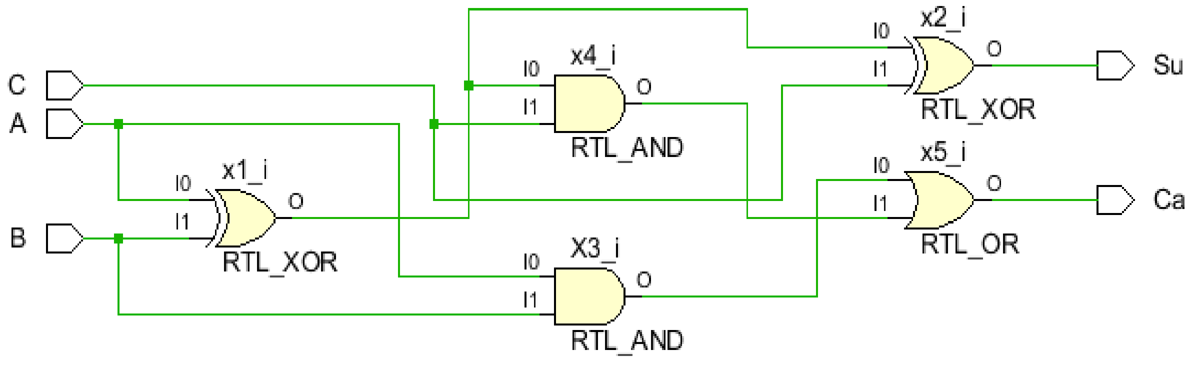


Figure 6.6: - SC of Full-Adder**.**

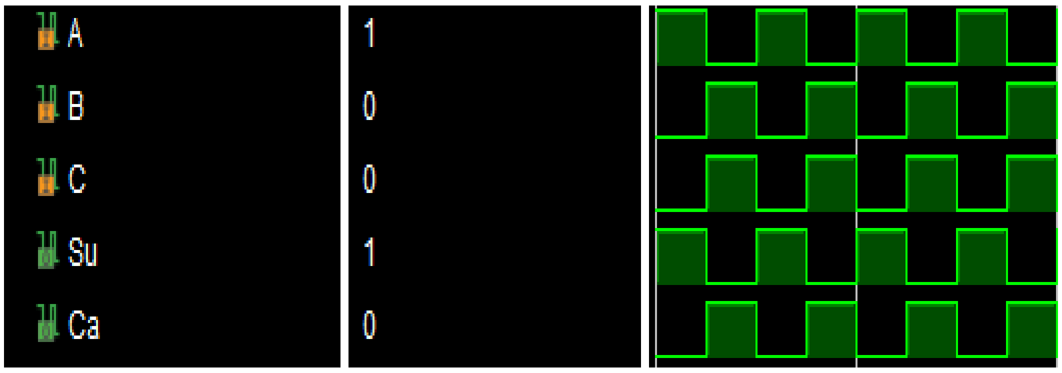


Figure 6.7: - Waveform of Full-Adder.

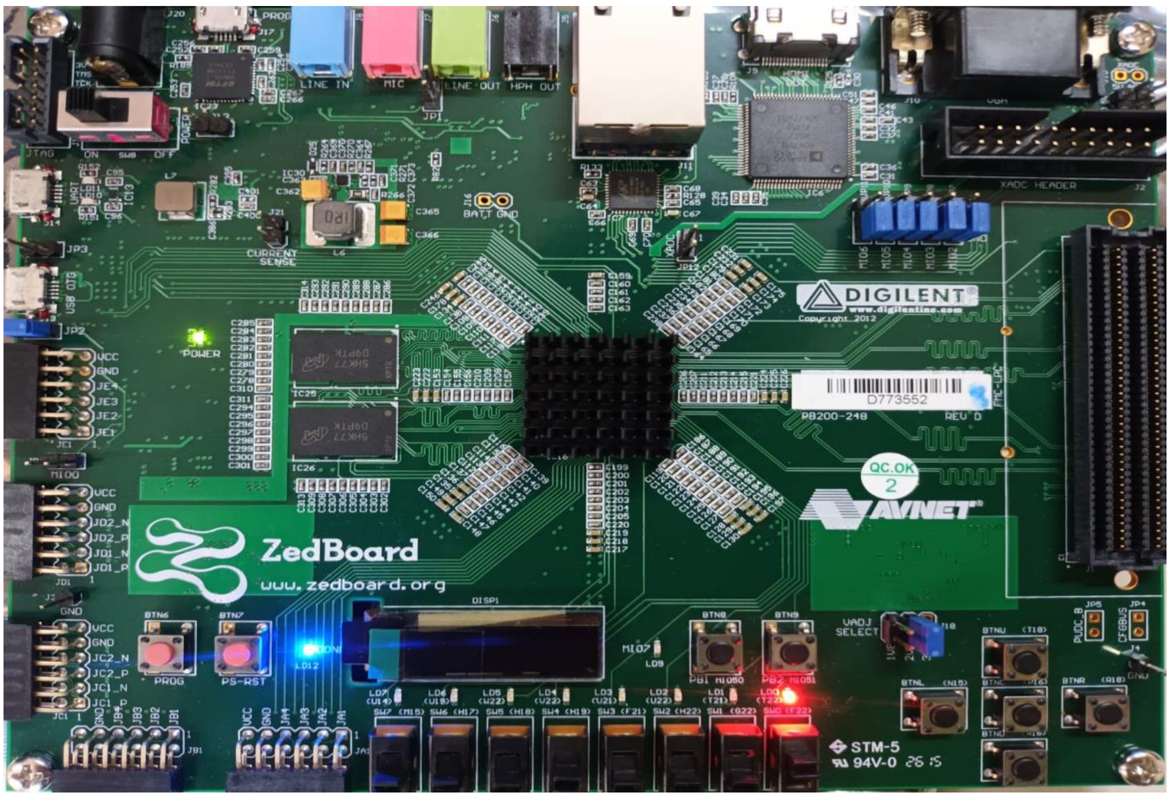


Figure 6.8: -FPGA implementation of Full-Adder

c) 4:2 Compressor

In binary arithmetic, a 4:2 compressor is a digital circuit that lowers the quantity of partial products in operations like multiplication. It streamlines the addition process in intricate arithmetic circuits by compressing four input bits into two output bits (a sum bit and a carry bit).

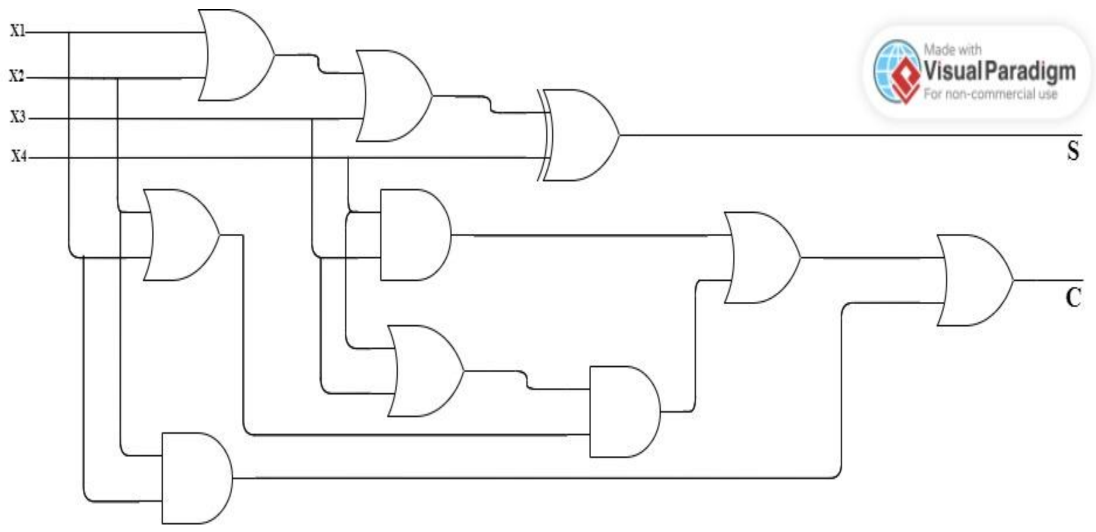


Figure 6.9: - Diagram of 4:2 Compressor

Table 6.3: Truth Table of 4:2 Compressor

S. No	X				S	V
	X1	X2	X3	X4	Su	Ca
0	0	0	0	0	0	0
1	0	0	0	1	1	0
2	0	0	1	0	1	0
3	0	0	1	1	0	1
4	0	1	0	0	1	0
5	0	1	0	1	0	1
6	0	1	1	0	0	1
7	0	1	1	1	1	1
8	1	0	0	0	1	0
9	1	0	0	1	0	1
10	1	0	1	0	0	1
11	1	0	1	1	1	1
12	1	1	0	0	0	1
13	1	1	0	1	1	1
14	1	1	1	0	1	1
15	1	1	1	1	1	1

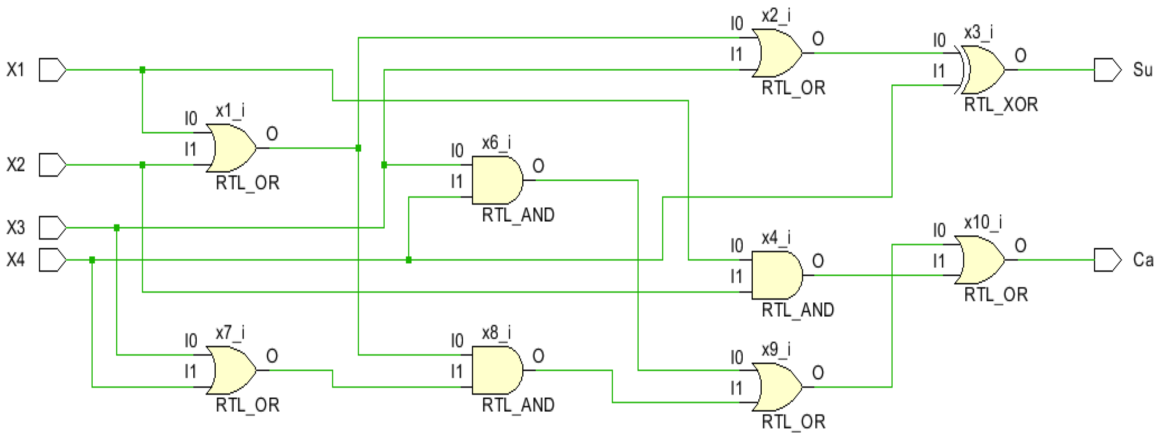


Figure 6.10: -SC of 4:2 Compressor

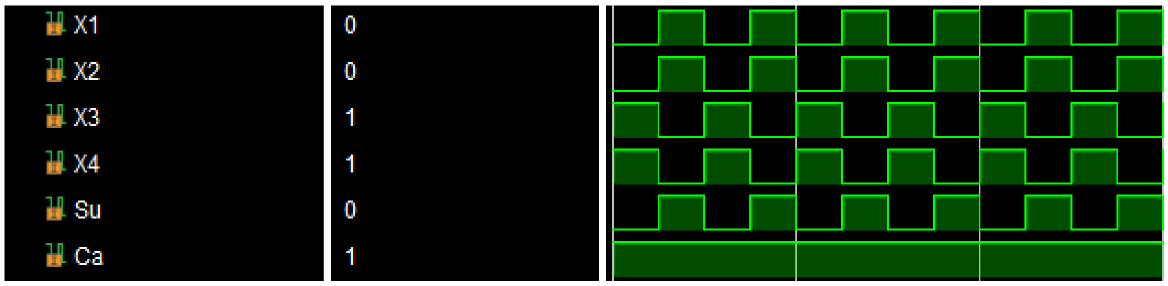


Figure 6.11: -Waveform of 4:2 Compressor

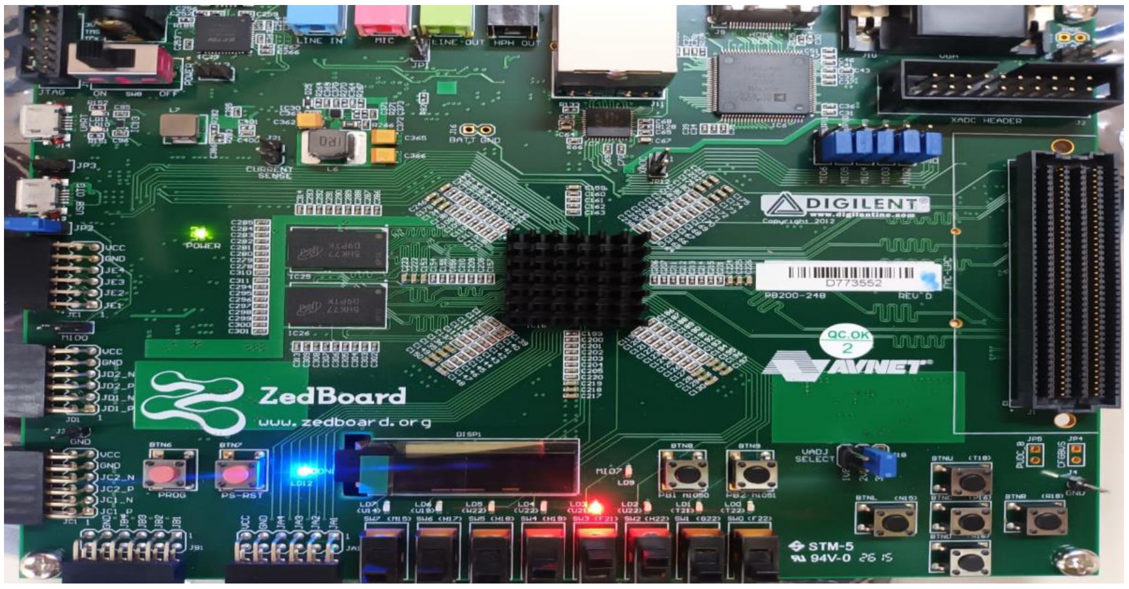


Figure 6.12: -FPGA Implementation of 4:2 Compressor

d) 4-bit Adder

A digital circuit that adds two 4-bit binary numbers is called a 4-bit adder. It is an essential part of digital electronics, especially processors and arithmetic logic units (ALUs). When binary addition is necessary in a variety of digital systems, such as in:

- Processors' Arithmetic Logic Units (ALUs).
- Digital Signal Processing (DSPs).
- Simple digital measuring devices and calculators.

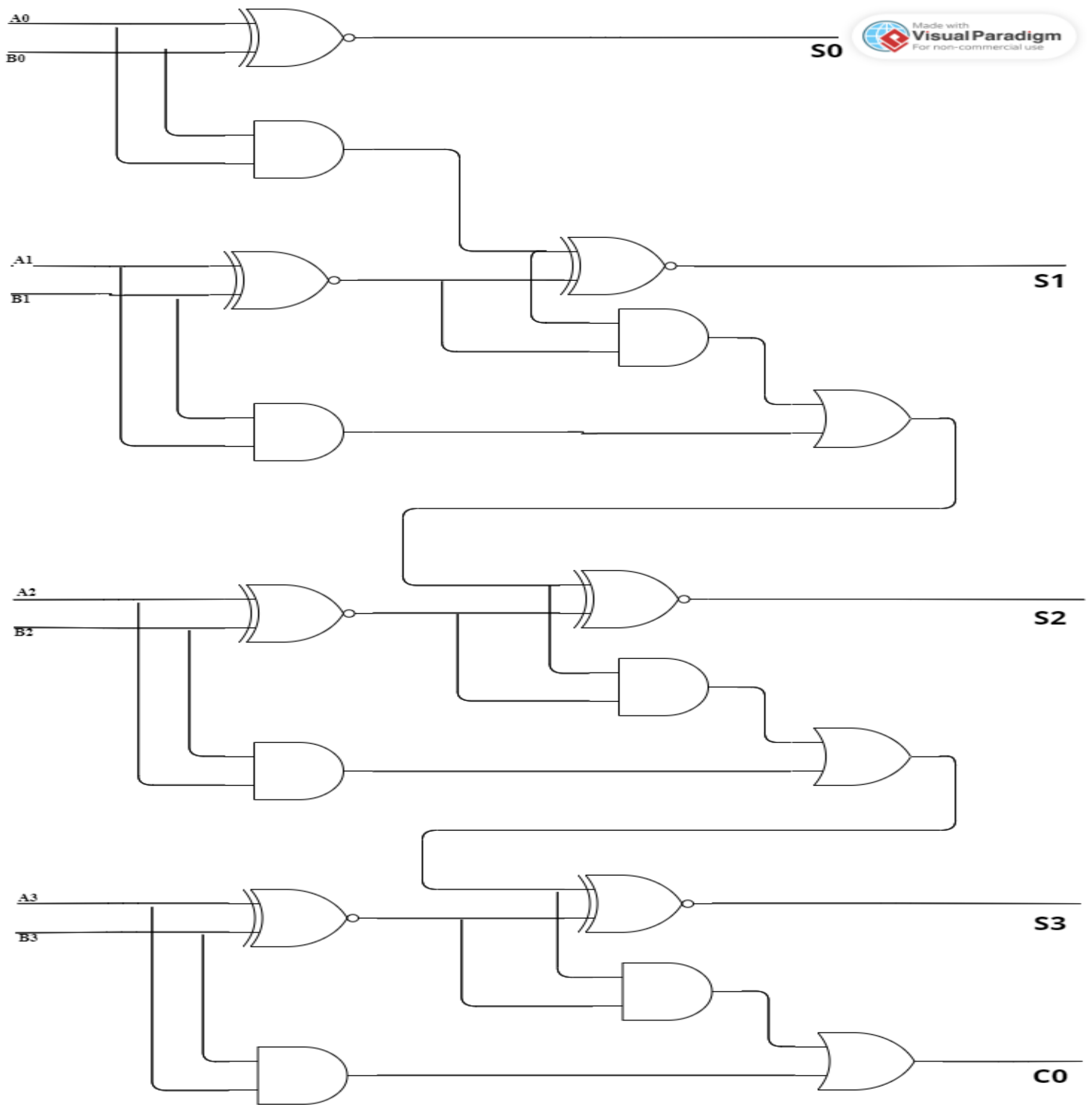


Figure 6.13: - Diagram of 4-bit Adder

TABLE6.4: - TRUTH TABLE OF 4-BIT ADDER

S.NO	A				B				SUM				CARRY
DECIMAL	A0	A1	A2	A3	B0	B1	B2	B3	S0	S1	S2	S3	C0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	0	0	0	1
2	0	0	1	0	0	0	1	0	0	0	0	1	0
3	0	0	1	1	0	0	1	1	0	0	0	1	1
4	0	1	0	0	0	1	0	0	0	0	1	0	0
5	0	1	0	1	0	1	0	1	0	0	1	0	1
6	0	1	1	0	0	1	1	0	0	0	1	1	0
7	0	1	1	1	0	1	1	1	0	0	1	1	1
8	1	0	0	0	1	0	0	0	0	1	0	0	0
9	1	0	0	1	1	0	0	1	0	1	0	0	1
10	1	0	1	0	1	0	1	0	0	1	0	1	0
11	1	0	1	1	1	0	1	1	0	1	0	1	1
12	1	1	0	0	1	1	0	0	0	1	1	0	0
13	1	1	0	1	1	1	0	1	0	1	1	0	1
14	1	1	1	0	1	1	1	0	0	1	1	1	0
15	1	1	1	1	1	1	1	1	0	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1	1	1	1

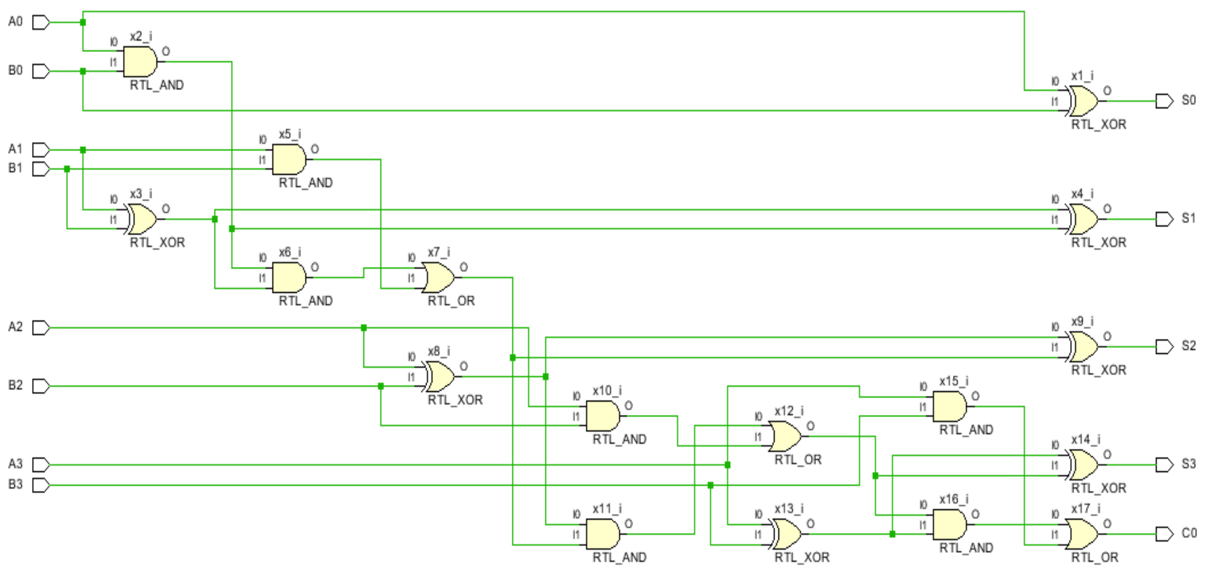


Figure 6.14: - SC of 4-bit Adder

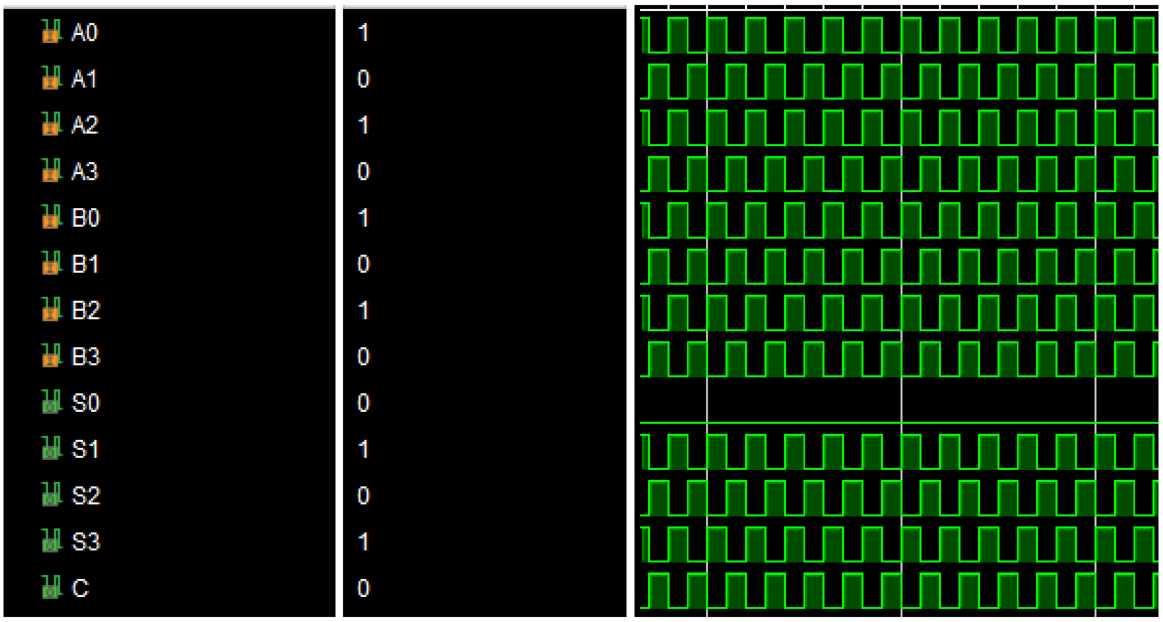


Figure 6.15: - Waveform of 4-bit Adder

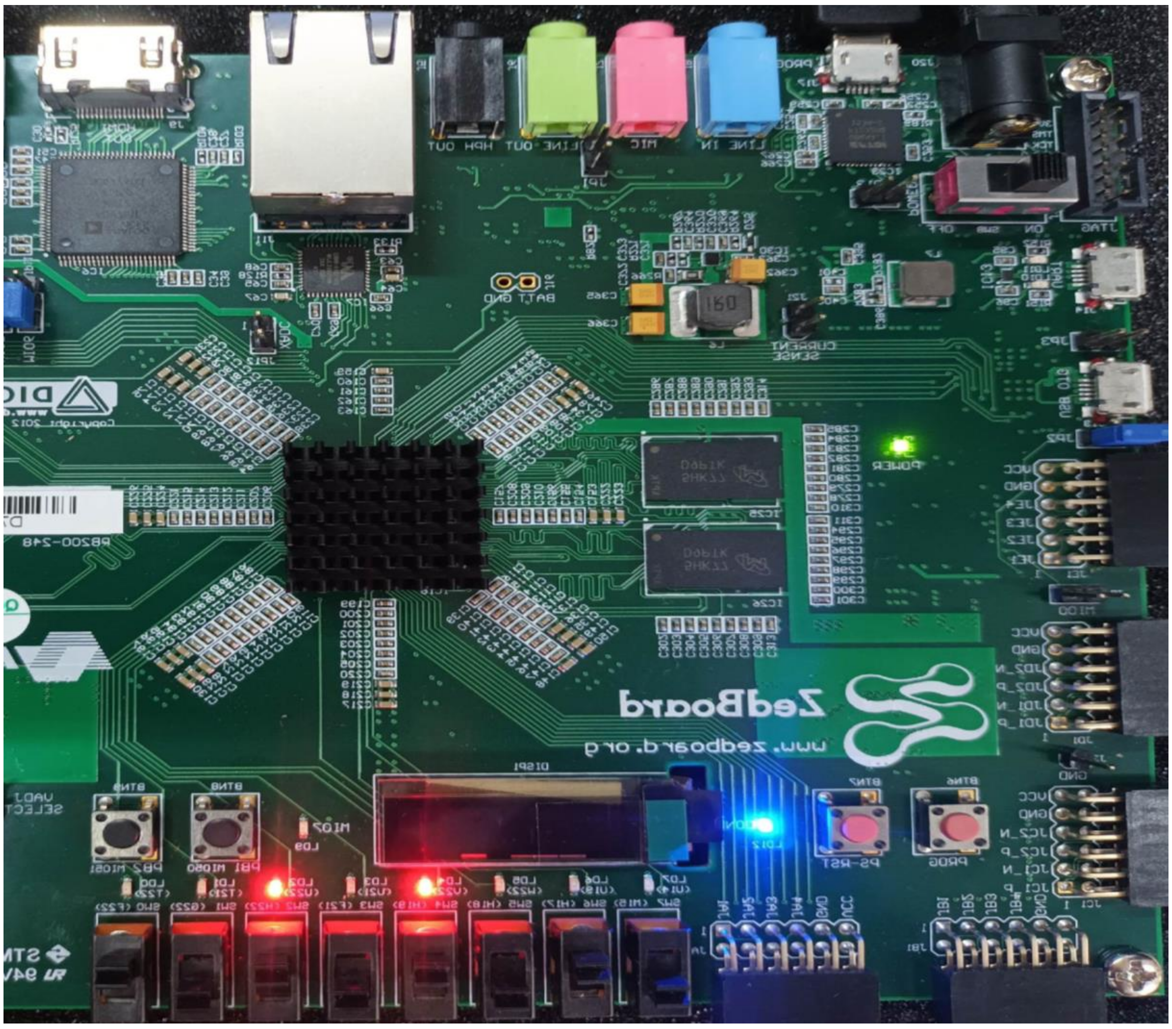


Figure 6.16: - FPGA implementation of 4-bit Adder

e) Proposed Wallace Tree Multiplier

Proposed Wallace Tree Multiplier consist of 4 Half-Adder, 2 4:2 compressor, 1 Full-Adder and 1 4-Bit Adder.

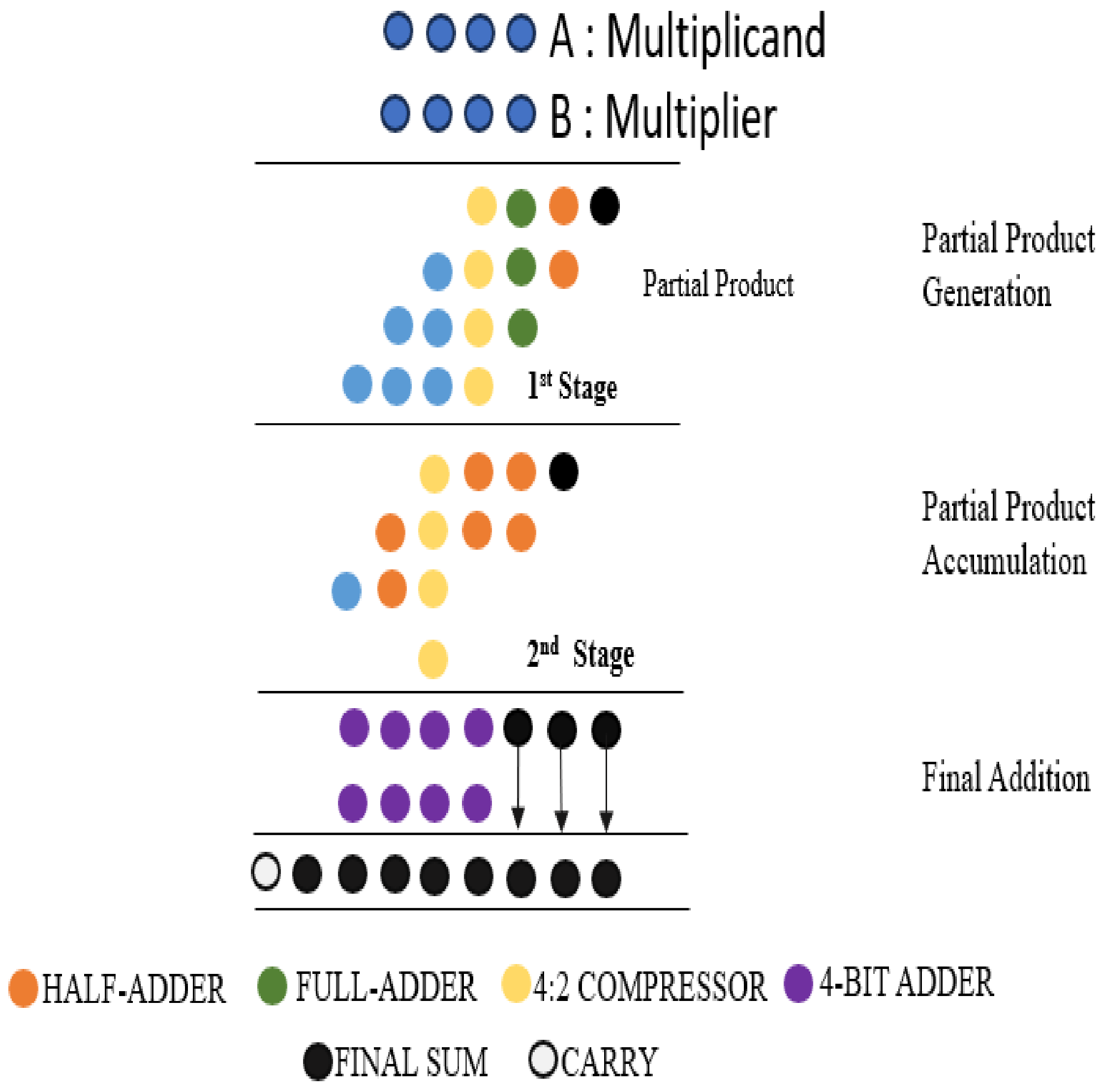


Figure 6.17: - Diagram of Proposed WTM

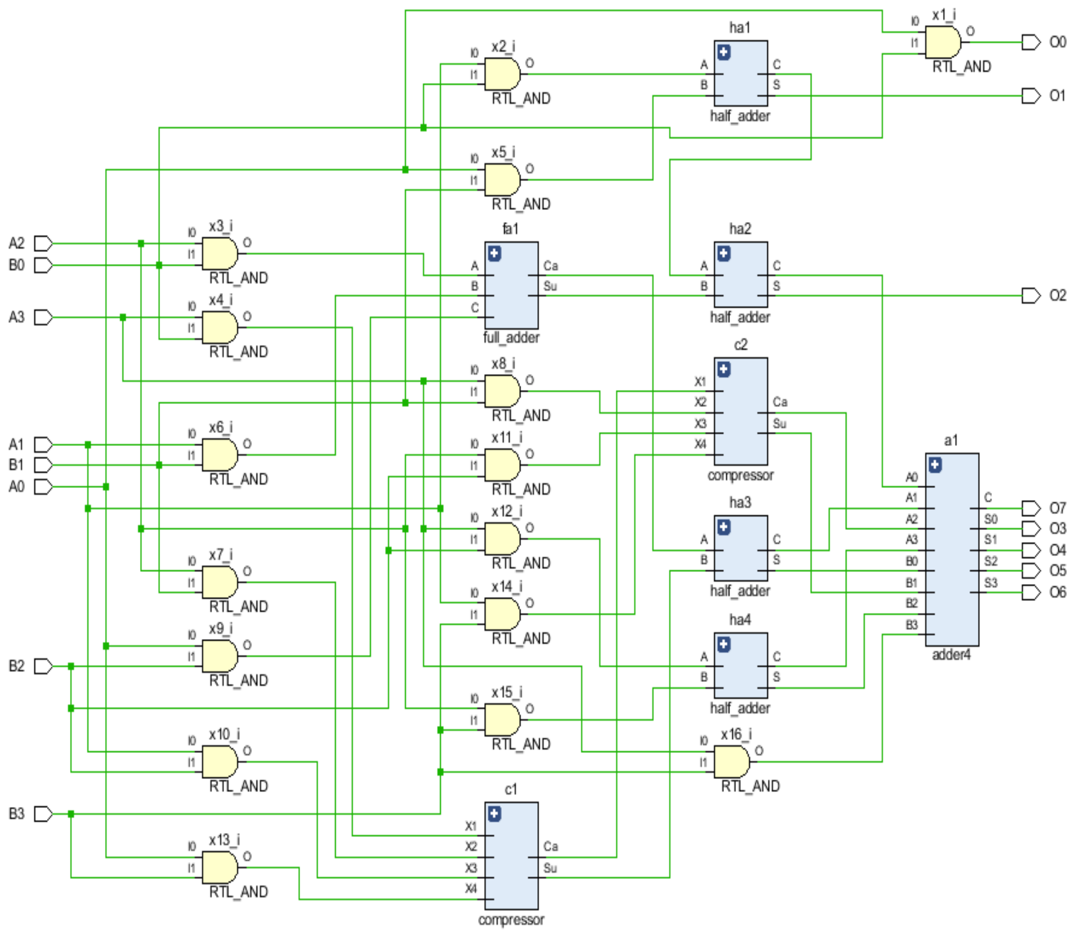
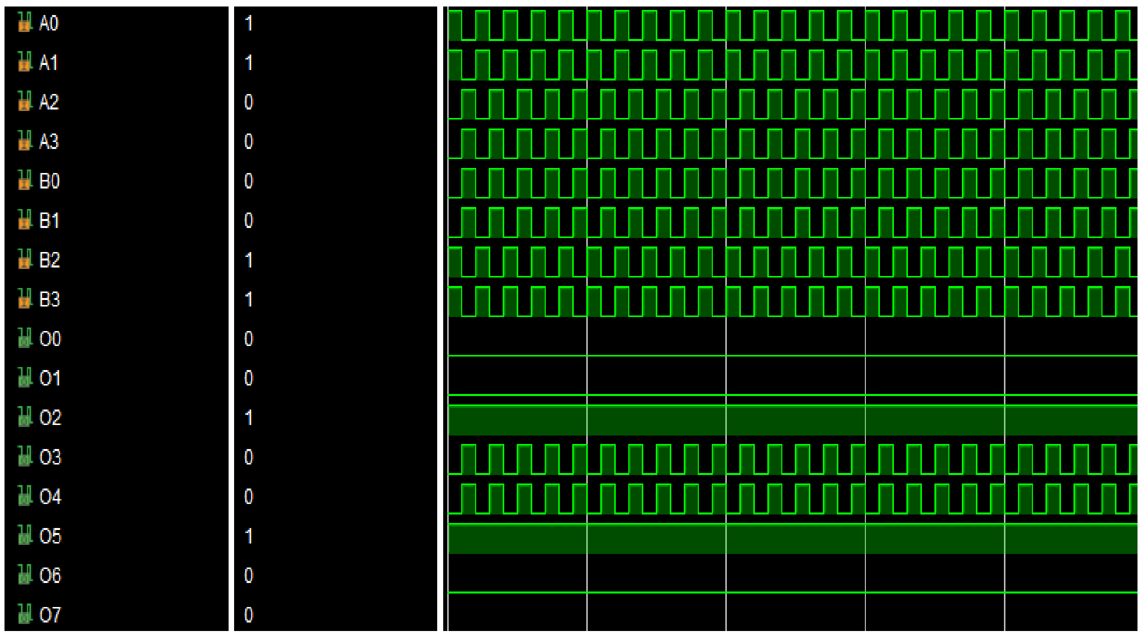


Figure 6.18: - SC of Proposed WTM



```
A0   1
A1   1
A2   0
A3   0
B0   0
B1   0
B2   1
B3   1
O0   0
O1   0
O2   1
O3   0
O4   0
O5   1
O6   0
O7   0
```

Figure 6.19: - Waveform of Proposed WTM

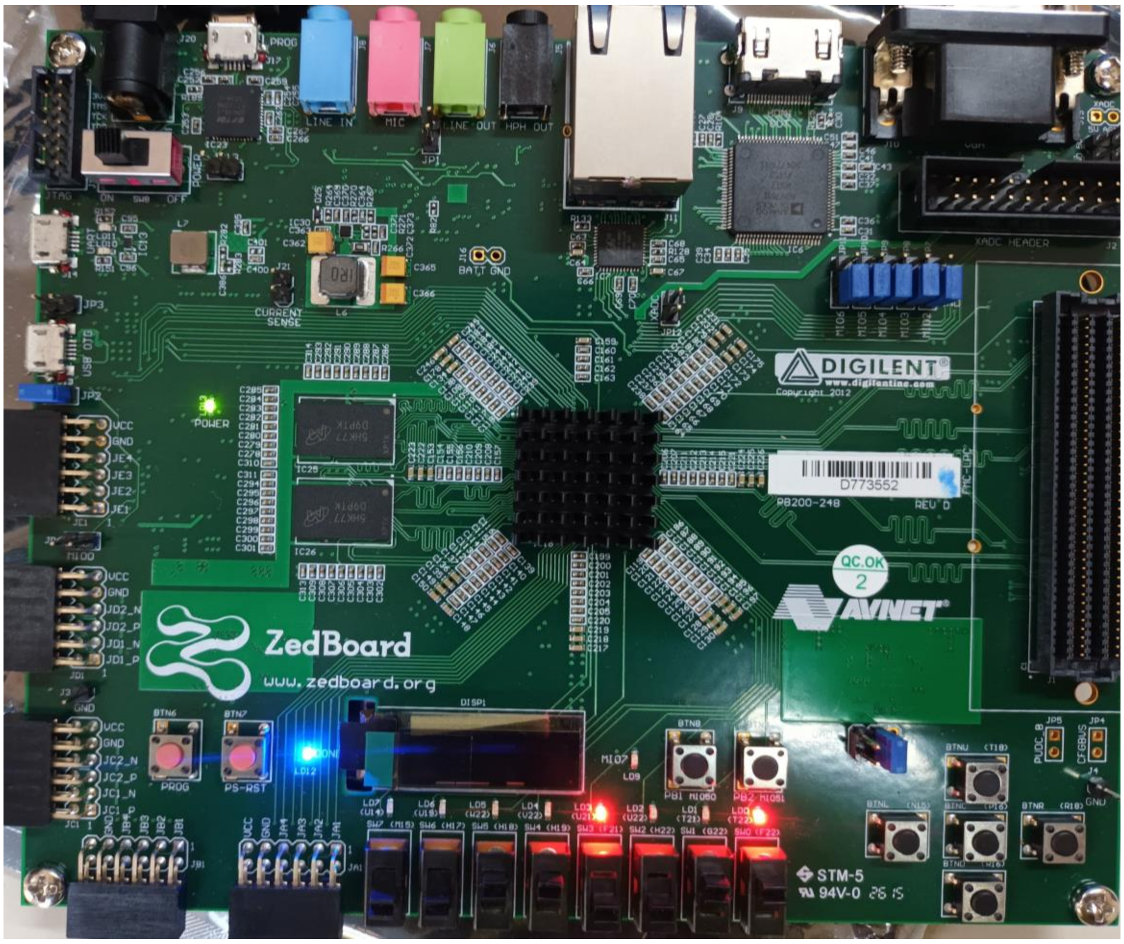


Figure 6.20: - FPGA implementation of Proposed WTM

Table 6.5: - Comparison of Different Parameters

PARAMETERS	WALLACE TREE	WALLACE TREE USING NOR	PROPOSED WALLACE TREE
No. of Slice	6	8	5
No. of LUT	21	22	17
No. of IOB	16	16	16
Power	3.684	2.422	2.021

The table shows that the Proposed WTM is best among the traditional WTM and WTM using NOR Logic.

Conclusion and Future Work

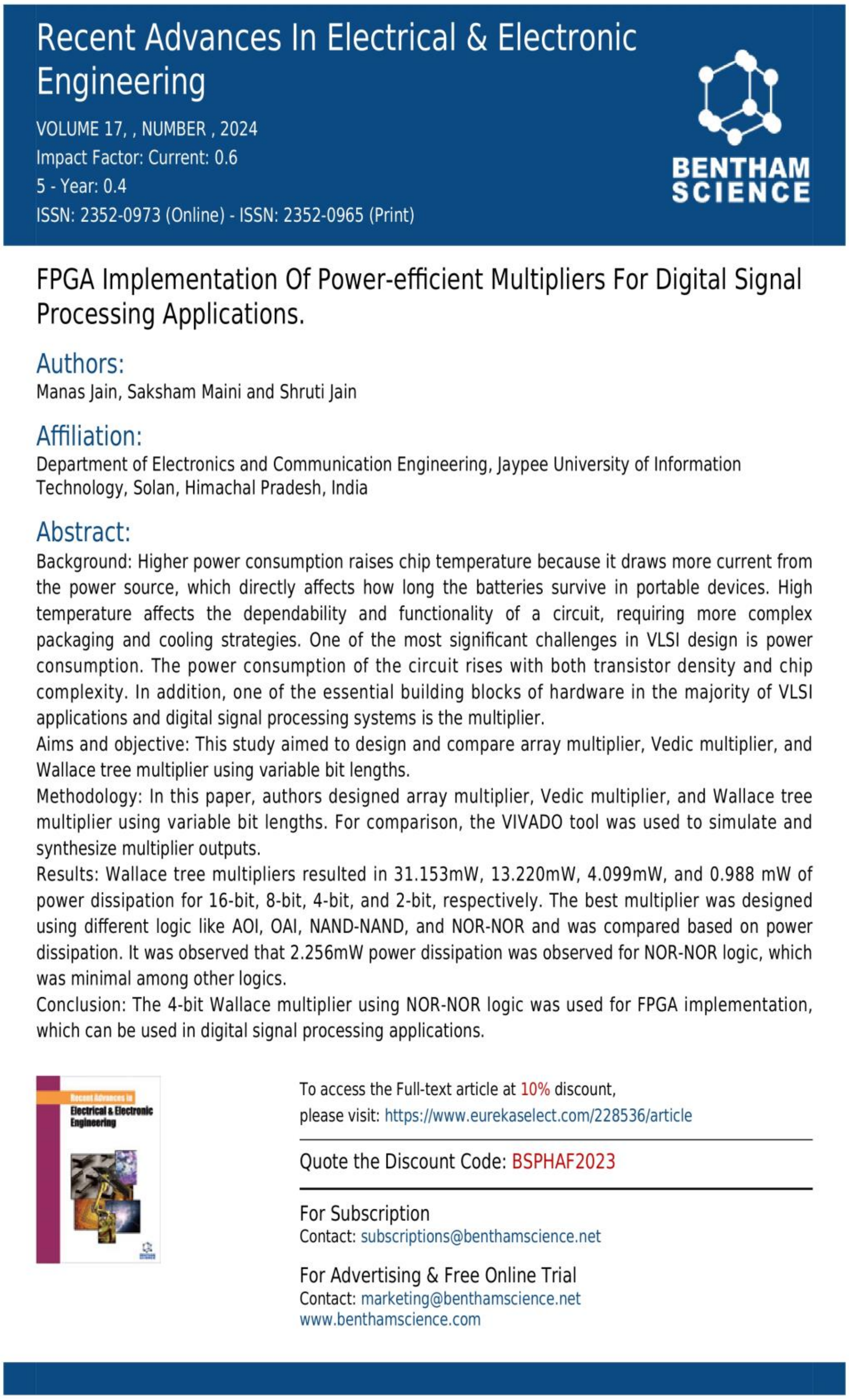
We have studied the VIVADO and Simulink and studied how to use Xilinx System Generator. We have successfully implemented the 2x2, 4x4, 8x8 and 16x16 bits with the help of Half-Adder, Full-Adder and Ripple Carry Adder. The multiplier architecture i.e. Array, Vedic and Wallace Tree are simulated using Vivado Design Suite. Synthesis and implementation are done using, ready-to-use digital circuit development platform based on the Zed Board Field Programmable Gate Array (FPGA). A comprehensive comparison based on specific parameters was conducted, leading to the determination that the Wallace tree multiplier outperformed the alternatives.

Also we performed the Wallace tree multiplier, considering four distinct gate types—NAND, NOR, AOI, and IOA. Through meticulous evaluation, it was established that the NOR gate configuration yielded the most optimal performance for the Wallace tree multiplier.

Further we design the New Wallace Tree Multiplier using half-adder, full-adder, 4:2 compressor and 4-bit adder and it is founded that the Proposed Wallace Tree multiplier is best among the Traditional Wallace Tree Multiplier and Wallace Tree Multiplier using the Nor-Logic

In future we will implement and simulate multiplier and will used circuit in different application.

LIST OF PUBLICATION



Recent Advances In Electrical & Electronic Engineering

VOLUME 17, , NUMBER , 2024

Impact Factor: Current: 0.6

5 - Year: 0.4

ISSN: 2352-0973 (Online) - ISSN: 2352-0965 (Print)

BENTHAM SCIENCE

FPGA Implementation Of Power-efficient Multipliers For Digital Signal Processing Applications.

Authors:

Manas Jain, Saksham Maini and Shruti Jain

Affiliation:

Department of Electronics and Communication Engineering, Jaypee University of Information Technology, Solan, Himachal Pradesh, India

Abstract:

Background: Higher power consumption raises chip temperature because it draws more current from the power source, which directly affects how long the batteries survive in portable devices. High temperature affects the dependability and functionality of a circuit, requiring more complex packaging and cooling strategies. One of the most significant challenges in VLSI design is power consumption. The power consumption of the circuit rises with both transistor density and chip complexity. In addition, one of the essential building blocks of hardware in the majority of VLSI applications and digital signal processing systems is the multiplier.

Aims and objective: This study aimed to design and compare array multiplier, Vedic multiplier, and Wallace tree multiplier using variable bit lengths.

Methodology: In this paper, authors designed array multiplier, Vedic multiplier, and Wallace tree multiplier using variable bit lengths. For comparison, the VIVADO tool was used to simulate and synthesize multiplier outputs.

Results: Wallace tree multipliers resulted in 31.153mW, 13.220mW, 4.099mW, and 0.988 mW of power dissipation for 16-bit, 8-bit, 4-bit, and 2-bit, respectively. The best multiplier was designed using different logic like AOI, OAI, NAND-NAND, and NOR-NOR and was compared based on power dissipation. It was observed that 2.256mW power dissipation was observed for NOR-NOR logic, which was minimal among other logics.

Conclusion: The 4-bit Wallace multiplier using NOR-NOR logic was used for FPGA implementation, which can be used in digital signal processing applications.

References

[1] https://www.geeksforgeeks.org/vlsi-design-cycle/

[2] https://technobyte.org/multiplier-2-bit-3-bit-digital/#:~:text=A%20multiplier%20is%20a%20combinational%20logic%20circuit%20that, as%20a%20binary%20multiplier%20or%20a%20digital%20multiplier.

[3] P. Martha, N. Kajal, P. Kumari and R. Rahul, "An efficient way of implementing high speed 4-Bit advanced multipliers in FPGA," *2018 2nd International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, Kolkata, India, 2018, pp. 1-5, doi: 10.1109/IEMENTECH.2018.8465375.

[4] https://www.slideshare.net/sabihasulthana9279/multipliers-in-vlsi

[5] https://www.quora.com/What-is-meant-by-FPGA-and-which-language-is-use-for-it

[6] U. A. Kumar, S. K. Chatterjee and S. E. Ahmed, "Low-Power Compressor-Based Approximate Multipliers With Error Correcting Module," in IEEE Embedded Systems Letters, vol. 14, no. 2, pp. 59-62, June 2022, doi: 10.1109/LES.2021.3113005.

[7] K. G. Hemamithra, S. Lakshmi Priya, K. Lakshmirajan, R. Mohanrai and S. R. Ramesh, "FPGA Implementation of Power Efficient Approximate Multipliers," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2018, pp. 1281-1285, doi: 10.1109/RTEICT42901.2018.9012325.

[8] F. Sabetzadeh, M. H. Moaiyeri and M. Ahmadinejad, "An Ultra-Efficient Approximate Multiplier With Error Compensation for Error-Resilient Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 70, no. 2, pp. 776-780, Feb. 2023, doi: 10.1109/TCSII.2022.3215065.

[9] A. G. M. Strollo, E. Napoli, D. De Caro, N. Petra, G. Saggese and G. Di Meo, "Approximate Multipliers Using Static Segmentation: Error Analysis and Improvements," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 6, pp. 2449-2462, June 2022, doi: 10.1109/TCSI.2022.3152921.

[10] I. Sayahi, M. Machhout and R. Tourki, "FPGA implementation of matrix-vector multiplication using Xilinx System Generator," 2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET), Hammamet, Tunisia, 2018, pp. 290-295, doi: 10.1109/ASET.2018.8379873.

[11] M. Ahmadinejad and M. H. Moaiyeri, "Energy- and Quality-Efficient Approximate Multipliers for Neural Network and Image Processing Applications," in IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 2, pp. 1105-1116, 1 April-June 2022, doi:

10.1109/TETC.2021.3072666.

[12] https://doi.org/10.1007/978-981-16-8550-7

[13] Ram, G.C., Subbarao, M.V., Kumar, D.G., Terlapu, S.K. (2022). FPGA Implementation of 16-Bit Wallace Multiplier Using HCA. In: Chakravarthy, V.V.S.S.S., Flores-Fuentes, W., Bhateja, V., Biswal, B. (eds) Advances in Micro-Electronics, Embedded Systems and IoT. Lecture Notes in Electrical Engineering, vol 838. Springer, Singapore.

[14] N. Van Toan and J. -G. Lee, "FPGA-Based Multi-Level Approximate Multipliers for High Performance Error-Resilient Applications," in IEEE Access, vol. 8, pp. 25481-25497, 2020, doi: 10.1109/ACCESS.2020.2970968

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 23/05/2024

Type of Document (Tick): PhD Thesis | M.Tech/M.Sc. Dissertation | [✓ B.Tech./B.Sc./BBA/Other]

Name: MANAS JAIN / SAKSHAM Department: E.C.E Enrolment No 201097/201004

Contact No. 9548466530/8837893168 E-mail. a.manasjain@gmail.com

Name of the Supervisor: Dr. SHRUTI JAIN

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): FPGA IMPLEMENTATION OF POWER-EFFICIENT MULTIPLIER

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages = 71
- Total No. of Preliminary pages = 12
- Total No. of pages accommodate bibliography/references = 59

[illegible]

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at 16 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

[illegible] 23/5/24

(Signature of Guide/Supervisor)

[illegible] 23/05/2024

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Abstract & Chapters Details	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Page counts	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete Thesis/Report in (PDF) & DOC (Word File) through your Supervisor/Guide at plagcheck.juit@gmail.com