PERFORMANCE COMPARISON REVIEW OF 8-3 COMPRESSOR ON FPGA

LEONG YUHAO

ELECTRICAL AND ELECTRONICS ENGINEERING

UNIVERSITI TEKNOLOGI PETRONAS

JANUARY 2017

Performance Comparison Review of 8-3 Compressor on FPGA

by

Leong Yuhao

17919

Dissertation submitted in partial fulfilment of

the requirements for the

Bachelor of Engineering (Hons)

(Electrical and Electronics)

JANUARY 2017

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

PERFORMANCE COMPARISON REVIEW OF 8-3 COMPRESSOR ON FIELD-PROGRAMMABLE GATE ARRAY (FPGA)

by

Leong Yuhao

17919

A project dissertation submitted to the

Electrical and Electronics Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfilment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL AND ELECTRONICS)

Approved by,

_____

(Lo Hai Hiung)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

January 2017

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

LEONG YUHAO

ABSTRACT

Compressors are commonly utilized in multipliers to cut down partial products in a parallel manner. Its main usage in this project is to go beyond the limit of fabrication technology to improve the computer performance. 2:1 compressors (half adder), 3:2 compressors (full adder), and 4:2 compressors should be well understood as it is the fundamental for building higher order compressors. In this paper, a 7-3, 8-3, 8-4, 9-3, and 9-4 researched compressors design either consisted of adder circuits or multiplexer circuits will be discussed and compared in terms of number of logic gates used, cell area and power delay product (PDP) in order to find the best method to implement the 8-3 compressor design.

ACKNOWLEDGEMENT

I am utilizing this utmost occasion to speak out my appreciation to all who has given me the comprehensive and continuous guidance throughout my Final Year Project period at Universiti Teknologi PETRONAS. This has given me a very praiseworthy moment which is hard to be found from anywhere else. I am grateful for their aspiring guidance, valuable constructive criticisms and kind advices during my project-doing period. The great experiences and skills shared by them give me a very new knowledge either from the aspect of organization or from the aspect of industrial world. To add-on, the practical project is beneficial for me in order to have better understandings on my role in Intel Penang.

A special thanks to my Final Year Project supervisor, Mr. Lo Hai Hiung for providing me a chance to be part of his project under his supervision. He gave me an interesting project titled "Performance Comparison Review of 8-3 Compressor on Filed-Programmable Gate Array (FPGA)" which covered working principles of compressor with the aids of Altera Quartus II Web Edition and Altera-ModelSim using Verilog Hardware Description Language and I feel honoured to be given this golden chance to learn new things. His willingness to share his experiences and prompt responses to my inquiries has tremendously given me a huge contribution in order for me to complete my final year project successfully.

Last but not least, an endeavour over a period is hard to be successful without the constructive advices, commitments, and supports from all my beloved friends, family, and my kind and helpful colleagues. Without their positive attitudes, moral supports, and friendly helps, I may not be able to complete my 8-months final year project to be such successful. Thanks for all the help! Be it little or huge, it definitely helped me a lot and I could not have done much without your cooperation!

TABLE OF CONTENTS

CERTIFICATE OF APPROVAL II

CERTIFICATE OF ORIGINALITY III

ABSTRACT IV

ACKNOWLEDGE V

LIST OF ABBREVIATION X

CHAPTER 1: INTRODUCTION 1
1.1 BACKGROUND 1
1.2 PROBLEM STATEMENT 2
1.3 OBJECTIVES 2
1.4 SCOPES OF STUDY 3

CHAPTER 2: LITERATURE REVIEW 4
2.1 LITERATURE REVIEWS 4
2.2 9-4 COMPRESSORS 5
2.3 8-4 COMPRESSORS 6
2.4 7-3 COMPRESSORS 8
2.5 8-3 COMPRESSORS 8
2.6 7-2 COMPRESSORS 10

CHAPTER 3: METHODOLOGY 12
3.1 RESEARCH METHODOLOGY 12
3.2 RESEARCH ACTIVITIES 14

CHAPTER 4: RESULTS AND DISCUSSIONS 19
4.1 8-3 COMPRESSOR (FULL AND HALF ADDERS) 19
4.2 8-4 COMPRESSOR (FULL AND HALF ADDERS) 23
4.3 9-4 COMPRESSOR (FULL AND HALF ADDERS) 27
4.4 9-3 COMPRESSOR (FULL AND HALF ADDERS) 32
4.5 7-3 COMPRESSOR (FULL AND HALF ADDERS) 36
4.6 7-3 COMPRESSOR (MUX AND XOR LOGIC GATES) 40
4.7 8-4 COMPRESSOR (MUX AND HALF ADDERS) 44
4.8 9-4 COMPRESSOR (MUX AND HALF ADDERS) 49
4.9 DISCUSSIONS 54

CHAPTER 5: CONCLUSION AND RECOMMENDATION 60
5.1 CONCLUSION 60
5.2 RECOMMENDATIONS 60

REFERENCES 61

APPENDIX 63

LIST OF FIGURES

Figure 1: Pictures showing Partial Product Generation and Final Product Computation 1
Figure 2: Implementation of 9-4 Compressor using Half and Full Adders [1] 6
Figure 3: Implementation of 8-4 Compressor using Multiplexer [1] 7
Figure 4: The Overall Proposed Structure of m:3 Compressor [3] 9
Figure 5: 8:3 New Hybrid Wide Compressor [15] 9
Figure 6: The Overall Proposed Structure of 7:2 Compressor [14] 11
Figure 7: Research Methodology Used in Project 12
Figure 8: Research Activities Used in Project 14
Figure 9: Minimum 8-3 Compressor Propagation Delay [12] 19
Figure 10: 8-3 Compressor Propagation Delay [12] 19
Figure 11: RTL View of 8-3 Compressor Model [12] 20
Figure 12: RTL View in Relationship with Input iSW[7] and Output oLEDG[1] [12] 20
Figure 13: 8-3 Compressor Functional Simulation using ModelSim-Altera 21
Figure 14: Error Result of 8-3 Compressor Functional Test bench on Console 21
Figure 15: Overall 8-3 Compressor Model Timing Simulation 22
Figure 16: Timing Simulation for iSW[0..7] with bit 1s and output oLEDG[0..2] 22
Figure 17: Minimum 8-4 Compressor Model Propagation Delay [1] 23
Figure 18: 8-4 Compressor Model Maximum Propagation Delay [1] 23
Figure 19: Overall RTL View of 8-4 Compressor Model [1] 24
Figure 20: RTL View in Relationship with Input iSW[0] and Output oLEDG[3] [1] 24
Figure 21: 8-4 Compressor Functional Simulation using ModelSim-Altera 25
Figure 22: Error Result of 8-4 Compressor Functional Test bench on Console 25
Figure 23: Overall 8-4 Compressor Model Timing Simulation 26
Figure 24: Timing Simulation for Inputs iSW[0..7] with bit 1 and output oLEDG[0..3] 26
Figure 25: Minimum 9-4 Compressor Model Propagation Delay [1] 27
Figure 26: 9-4 Compressor Model Maximum Propagation Delay [1] 28
Figure 27: Overall RTL View of 9-4 Compressor Model [1] 29
Figure 28: RTL View in Relationship with Input iSW[7] and Output oLEDG[3] [1] 29
Figure 29: 9-4 Compressor Functional Simulation using ModelSim-Altera 30
Figure 30: Error Result of 9-4 Compressor Functional Test bench on Console 30
Figure 31: Overall 9-4 Compressor Model Timing Simulation 31
Figure 32: Timing Simulation for Inputs iSW[0..8] with bit 1s and output oLEDG[0..3] 31
Figure 33: Minimum 9-3 Compressor Model Propagation Delay [12] 32
Figure 34: 9-3 Compressor Model Maximum Propagation Delay [12] 32
Figure 35: Overall RTL View of 9-3 Compressor Model [12] 33
Figure 36: RTL View in Relationship with Input iSW[7] and Output oLEDG[1] [12] 33
Figure 37: 9-3 Compressor Functional Simulation using ModelSim-Altera 34
Figure 38: Error Result of 9-3 Compressor Functional Test bench on Console 34
Figure 39: Overall 9-3 Compressor Model Timing Simulation 35
Figure 40: Timing Simulation for Inputs iSW[0..8] with bit 1s and output oLEDG[0..3] 35
Figure 41: Minimum 7-3 Compressor Model Minimum Propagation Delay [10] 36
Figure 42: 7-3 Compressor Model Maximum Propagation Delay [10] 36
Figure 43: Overall RTL View of 7-3 Compressor Model [10] 37
Figure 44: RTL View in Relationship with Input iSW[5] and Output oLEDG[1] [10] 37
Figure 45: 7-3 Compressor Functional Simulation using ModelSim-Altera 38
Figure 46: Error Result of 7-3 Compressor Functional Test bench on Console 38

Figure 47: Overall 7-3 Compressor Model Timing Simulation 39
Figure 48: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[1] with bit 1 39
Figure 49: Minimum 7-3 Compressor Model Minimum Propagation Delay using MUX and XOR Logic Gates [10] 40
Figure 50: 7-3 Compressor Model Maximum Propagation Delay using MUX and XOR Logic Gates [10] 40
Figure 51: Overall RTL View of MUX-XOR Logic Gates 7-3 Compressor Model [10] 41
Figure 52: RTL View in Relationship with Input iSW[5] and Output oLEDG[1] [10] 41
Figure 53: 7-3 Compressor Functional Simulation using ModelSim-Altera 42
Figure 54: Error Result of 7-3 Compressor Functional Test bench on Console 42
Figure 55: Overall 7-3 MUX-XOR Compressor Model Timing Simulation 43
Figure 56: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[1] with bit 1 43
Figure 57: Minimum 8-4 Compressor Model Minimum Propagation Delay using MUX and Half Adders [1] 44
Figure 58: 8-4 Compressor Model Maximum Propagation Delay using MUX and Half Adders [1] 45
Figure 59: Overall RTL View of MUX and Half Adders' 8-4 Compressor Model [1] 46
Figure 60: RTL View in Relationship with Input iSW[2] and Output oLEDG[3] [1] 46
Figure 61: 8-4 Compressor Functional Simulation using ModelSim-Altera 47
Figure 62: Error Result of 8-4 Compressor Functional Test bench 47
Figure 63: Overall 8-4 MUX-Half Adder Compressor Model Timing Simulation 48
Figure 64: Timing Simulation for Inputs iSW[2] with bit 1 and output oLEDG[3] with bit 1 48
Figure 65: Minimum 9-4 Compressor Model Minimum Propagation Delay using MUX and Half Adder [1] 49
Figure 66: 9-4 Compressor Model Maximum Propagation Delay using MUX and Half Adder [1] 50
Figure 67: Overall RTL View of MUX-Half Adder 9-4 Compressor Model [1] 51
Figure 68: RTL View in Relationship with Input iSW[5] and Output oLEDG[3] [1] 51
Figure 69: 9-4 Compressor Functional Simulation using ModelSim-Altera 52
Figure 70: Error Result of 9-4 Compressor Functional Test bench 52
Figure 71: Overall 9-4 MUX-Half Adder Compressor Model Timing Simulation 52
Figure 72: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[3] with bit 1 53
Figure 73: Figure showing Simplified Concept of using 8-3 Compressor 54
Figure 74: 8-3 Compressor Model / Finite State Machine 63
Figure 75: 8-3 Compressor Functional Test bench 63
Figure 76: 8-4 Compressor Model / Finite State Machine 64
Figure 77: 8-4 Compressor Functional Test bench 64
Figure 78: 9-4 Compressor Model / Finite State Machine 65
Figure 79: 9-4 Compressor Functional Test bench 65
Figure 80: 9-3 Compressor Model / Finite State Machine 66
Figure 81: 9-3 Compressor Functional Test bench 66
Figure 82: 7-3 Compressor Model / Finite-State Machine 67
Figure 83: 7-3 Compressor Functional Test bench 67
Figure 84: 7-3 Compressor Model / Finite-State Machine using MUX-XOR Logic Gates 68

Figure 85: MUX-XOR Logic Gates 7-3 Compressor Functional Test bench..... 68
Figure 86: 8-4 Compressor Model / Finite-State Machine using MUX and Half Adders 69
Figure 87: MUX and Half Adders 8-4 Compressor Functional Test bench 69
Figure 88: 9-4 Compressor Model / Finite-State Machine using MUX and Half Adders 70
Figure 89: MUX and Half Adders 9-4 Compressor Functional Test bench 70

LIST OF TABLES

Table 1: Performance Comparison of 8-4 Compressor and 9-4 Compressor [1]..... 6
Table 2: Performance Comparison of 8-4 Compressor [1]..... 7
Table 3: Implementation Results with Final Ripple Adder [3]..... 10
Table 4: Evaluation of Proposed 7:2 Compressor with respect to Conventional 7:2 Compressor [14]..... 11
Table 5: Gantt chart for Final Year Project I 17
Table 6: Gantt chart for Final Year Project II..... 18
Table 7: Propagation Delay of input iSW[0..7] with Input 1s 22
Table 8: Propagation Delay of iSW[0..7] with Input 1s 26
Table 9: Propagation Delay of iSW[0..8] with Input 1s 31
Table 10: Propagation Delay of iSW[0..8] with Input 1s 35
Table 11: Propagation Delay of iSW[5] and oLEDG[1] with Input 1s 39
Table 12: Propagation Delay of iSW[5] and oLEDG[1] with Input 1s 43
Table 13: Propagation Delay of iSW[2] and oLEDG[3] with Input 1s 48
Table 14: Propagation Delay of iSW[5] and oLEDG[3] with Input 1s 53
Table 15: Summary of Overall Compressor Types in Relationship with Logic Gate Numbers and Critical Path Propagation Delay..... 55
Table 16: Summary of Overall Compressor Types in Relationship with Cell Area, Power Consumption and Power Delay Product (PDP) 55
Table 17: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 7-3 Compressors..... 56
Table 18: Cell Area, Power Consumption and Power Delay Product Comparisons between 7-3 Compressors..... 56
Table 19: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 8-4 Compressors..... 57
Table 20: Cell Area, Power Consumption and Power Delay Product Comparisons between 8-4 Compressors..... 57
Table 21: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 9-4 Compressors..... 58
Table 22: Cell Area, Power Consumption and Power Delay Product Comparisons between 9-4 Compressors..... 58

LIST OF ABBREVIATION

16-T : 16 Transistor

EDP : Energy Delay Product

PDP : Power Delay Product

VLSI : Very Large Scale Integration

CPU : Central Processing Unit

FPGA : Field-Programmable Gate Array

RTL : Register Transfer Language

HDL : Hardware Description Language

CMOS : Complementary Metal Oxide Semiconductor

CHAPTER 1: INTRODUCTION

1.1 BACKGROUND

Multiplication can be considered as a complicated and time-consuming arithmetic operation and it is the key operation in most of the signal processing algorithm [1] [2] [3] [4]. Designing good multipliers is challenging for Very Large Scale Integration (VLSI) as they normally have long latency, large area, and consuming significant power [1] [5]. A good multiplier must be physically compact, consuming low power, and having a good speed [1]. Generally, the process of multiplication can be split into three stages: generating partial product stage, reducing partial product stage, and computing final product stage [2] [3] [6]. The partial product generation and final product computation concept is shown in Figure 1. In VLSI circuit, the stage of reducing partial product will greatly reduce multiplier performance from the aspect of dissipating power and speed [5] [7] [8] [9]. A long critical vertical path during addition causes partial product reduction to have high latencies [10] [11]. Adders that are generally used to shorten the vertical critical path will cause issues such as uneven signal transitions and glitches and more number of adder stages are required to reduce the partial product [1] [8] [12]. To handle these issues, compressors are required to be utilized in multiplier design as it provides a regular structure in the stage of reducing partial product [5].

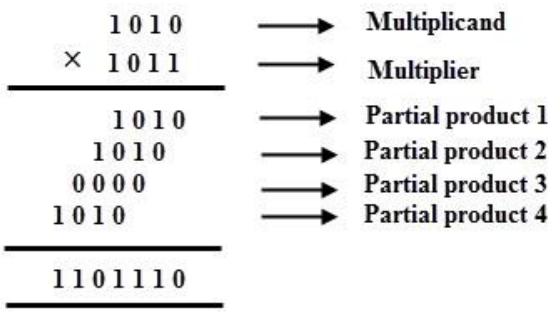


Figure 1: Partial Product Generation and Final Product Computation

1.2 PROBLEM STATEMENT

Computer system performance has flattened out as fabrication technology is reaching its physical limit. Thus, reviewing the circuit design is needed to search for possible speed up from smarter circuit design to give a faster computer for future demand. Typically, compressors are used in high speed addition and multiplication unit design in microprocessor. Thus, a faster compressor would allow for a faster addition and multiplication for future computer architecture systems design.

1.3 OBJECTIVES

The objectives of the project are:

- To repeat 7-3, 7-4, 8-3, 8-4, 9-3, and 9-4 compressor designs of others
- To verify researched compressor designs performance report
- To determine the best method of designing 8-3 compressor

1.4 SCOPES OF STUDY

This study was conducted by Leong Yuhao at Universiti Teknologi PETRONAS and it is vital as compressors consisted of adders and multiplexers and these are the fundamental for Central Processing Unit (CPU) to process data.

Besides that, this study also focuses on reading and understanding several research papers related to high order compressors such as 7-3, 7-4, 8-3, 8-4, 9-3, and 9-4 compressor designs. Then, the best way of designing the 8-3 compressor will be well-analysed by listing out the advantages and disadvantages of the compressors from the aspect of power consumption, propagation delay, and cell area and the best method of designing 8-3 compressor either using adders' method, combination of multiplexers and XOR logic gates, or combination of multiplexer and adders will be decided.

This project is relevant to Electrical and Electronics Engineering Programme as it allows me to write Verilog HDL and read signal transition diagrams for a better understanding of logic gates. Moreover, this project is feasible as it can be done with the aids of FPGA and Altera software for functional and timing simulation purposes.

CHAPTER 2: LITERATURE REVIEW

2.1 LITERATURE REVIEWS

Field-Programmable Gate Array (FPGA) is an integrated circuit containing programmable logic blocks arrays designed to be easily configured to logic gates to perform complex combinational functions, or merely simple logic gates like AND or XOR [13].

During partial product stage, multipliers need large amount of delay and power [10] [14]. At this stage, compressors will come into play where its usage is to reduce the partial product reduction. A compressor is merely an adder circuit which is a combinatorial device mainly utilized in multipliers to cut down the operands while summing partial products terms [10] [11]. It generates some sum signals by adding a number of equally-weighted bits. It is generally known to be used with the goal of cutting down a large number of inputs to a smaller number of outputs by utilising accumulation method in a parallel manner [12] [14]. Its primary usage is to concurrently sum up the large number of partial products happening in multipliers [10] [12] [14]. Compressors usage not only limited to reducing the vertical critical path of circuit but also simultaneously decreasing the stage operations [3] [7] [8]. The compressor structure avoids carry propagation. For instance, in 4:2 compressors, the value of C_{out} depends only on input A, B, C, and D but independent of C_{in}.

In general, a compressor will consist of 2 types of adders which are half adders and full adders [1] [2] [3] [5]. Half adders can be constructed easily with one AND and one Exclusive-OR (XOR) gate while full adders can be easily constructed using one OR, 2 AND, and 2 XOR gates [2]. The full adder's equation can be summarised as below:

$$Sum = (A \oplus B) \oplus Cin \qquad [2]$$

$$Cout = AB + Cin(A \oplus B) \qquad [2]$$

To have better understanding of higher order compressors, 4 types of compressors that are 8:4 compressors, 8:3 compressors, 7:3 compressors, and 7:2 compressors are being studied.

2.2 9-4 COMPRESSORS

R.Marimuthu, et al. [1] proposed an 9-4 compressor using 5 Full Adders and 2 half adder as shown in Figure 2. Its implementation results are shown in Table 1 which it was compared with the 8-4 compressor. From the table, the power and EDP of 9-4 compressor is higher than 8-4 compressor. It shows that the higher order compressors consume more power, longer critical path propagation delay and larger compressor cell area. It shows that the vertical critical path of 9-4 compressors is longer than 8-4 compressors as 9-4 compressor involved more logic gates which are 2 AND gates and 1 XOR gate more than the 8-4 compressor. By increasing the number of logic gates in higher order compressors, the critical path propagation delay of the compressors is increased by 5.66% , contributing to higher power consumed and larger area of cell.

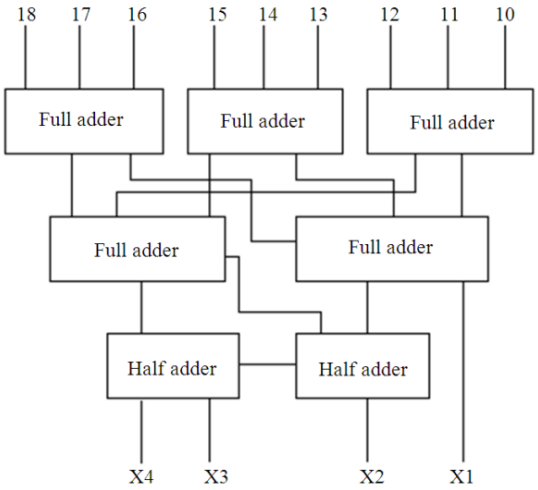


Figure 2: Implementation of 9-4 Compressor using Half and Full Adders [1]

Table 1: Performance Comparison of 8-4 Compressor and 9-4 Compressor [1]

	8-4 Compressor	9-4 Compressor
Parameters	Using full and half adder	Using full and half adder
Power (nW)	18276-250	21848.2180
Delay (ps)	1028.000	1082.0000
Area (µm2)	93.139	102.1900
EDP 10^{-24} (J-s)	19.310	25.5779

2.3 8-4 COMPRESSORS

R.Marimuthu, et al. [1] proposed an 8-4 compressor using 10 multiplexers and 1 half adder as shown in Figure 3. Its implementation results are shown in Table 2 which it was compared with the conventional compressor. From the table, the power and EDP of 8-4 compressor using multiplexer is lower than the 8-4 compressor using adders. It shows that the compressors using multiplexers is more energy and power efficient. However, the critical path propagation delay and cell area of compressor using multiplexer is larger than compressor using half and full adder. It shows that the vertical critical path of compressors using multiplexer is 4.18% longer than compressors using half and full adder.

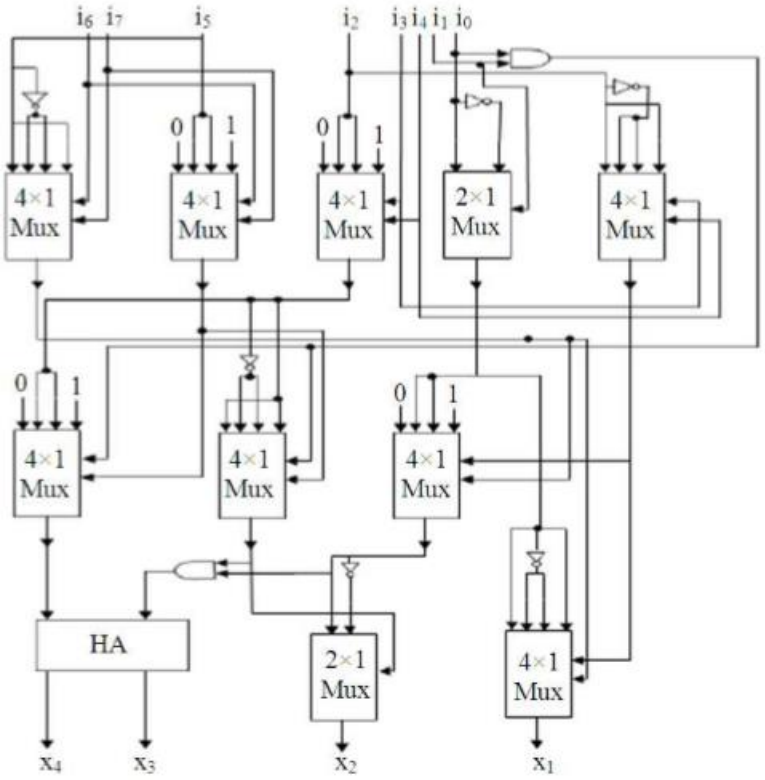


Figure 3: Implementation of 8-4 Compressor using Multiplexer [1]

Table 2: Performance Comparison of 8-4 Compressor [1]

	8-4 Compressor	
Parameters	Using full and half adder	Using multiplexer
Power (nW)	18276-250	15263.603
Delay (ps)	1028.000	1071.000
Area (μm2)	93.139	110.779
EDP 10^{-24} (J-s)	19.310	17.500

2.4 7-3 COMPRESSORS

Nirlakalla et al. [2] proposed a (7:3) compressor consisted of 4 full adders. 5 types of adders such as Transmission Gate (TG) CMOS, Transmission Function Full Adder (TFA), 16 Transistors (16-T), 14 Transistors (14-T), and Sense Energy Recovery Full Adder 10 Transistors (SERF 10-T) are being compared for (4:3), (5:3), (6:3), and (7:3) compressors. At gate level, via utilizing Xilinx ISE 9.1 synthesis tool simulations, 16-T full adder showed lowest PDP and EDP proving that it is the most energy efficient if compared to the other 4 types of adders mentioned. However, Nirlakalla et al. 2011 research has limitations as it does not consider the aspects of fan-in and area and these 2 aspects are vital since the fan-in of logic gates or area of compressors increase, the costs increase.

2.5 8-3 COMPRESSORS

Dandapat et al. [3] proposed an 8:3 compressor design based on Figure 4 general structure of m:3 compressors. It consisted of 5 full adders and 2 half adders. Its implementation result with final ripple adder is stated in Table 3 which having lower data arrival time, lower power consumption, lower PDP and fewer connections than 8:2 compressors. However, it has larger total cell area which will be the factor of limiting the features of m:3 compressors within a specific area. Based on Figure 4, the formula can be derived as below:

$$\sum_{i=1}^{3} Output + \sum_{i=1}^{p} Output\ Carrier = \sum_{i=1}^{m} Partial\ Product + \sum_{i=1}^{p} Input\ Carrier \qquad [3]$$

$$2^k + 2^{k+1} + 2^{k+2} + px2^{k+2} = (m+p)x2^k \qquad [3]$$

$$p = \left\lceil \frac{m-7}{3} \right\rceil, p \geq 0 \qquad [3]$$

Since $p = \frac{1}{3} > 0$ when m = 8, therefore, m = 1 (get ceiling)

However, Dandapat et al. does not consider the fan-in of logic gate. It is vital as the more fan-in of logic gate needed, the higher costs and higher complexity it is.

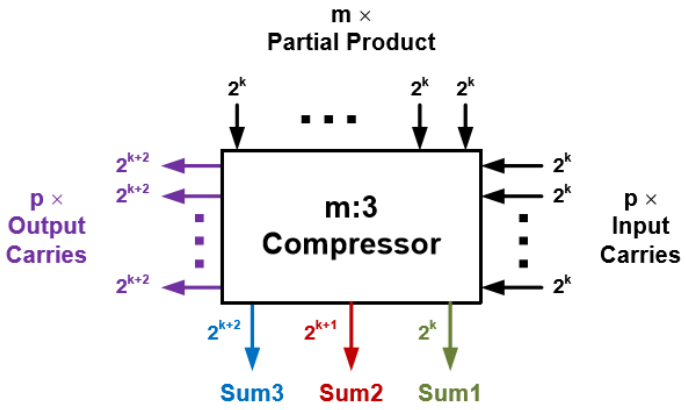


Figure 4: The Overall Proposed Structure of m:3 Compressor [3]

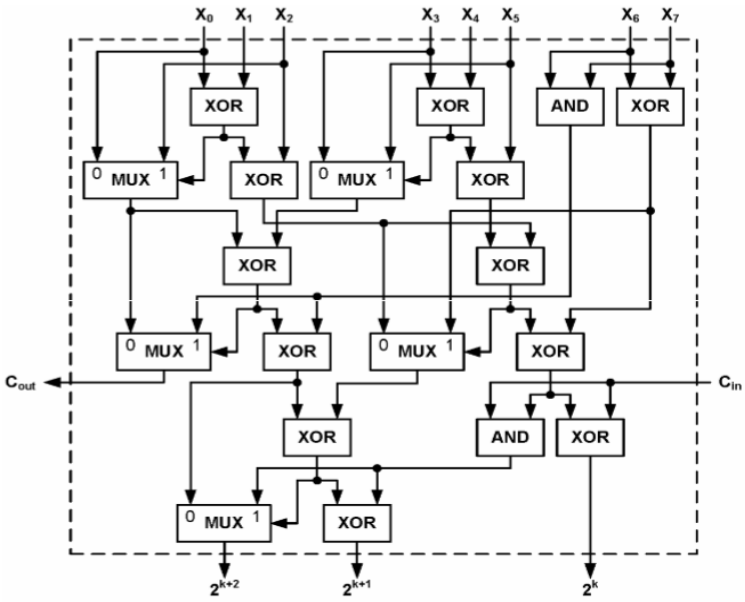


Figure 5: 8:3 New Hybrid Wide Compressor [15]

Table 3: Implementation Results with Final Ripple Adder [3]

Multiplier	by only *m*:2 Compressors	by only *m*:3 Compressors
Data Arrival Time	4.69 nsec	4.46 nsec
Total Cell Area	4340.9 μm^2	4428.4 μm^2
Cell Internal Power	1079.3 µW	1002.9 µW
Net Switching Power	542.16 µW	462.22 µW
Total Dynamic Power	1621.4 µW	1465.2 µW
Cell Leakage Power	8.4090 µW	8.4990 µW

2.6 7-2 COMPRESSORS

Rouholamini et. al. [14] proposed a 7:2 compressor consisted of 5 pieces of 3:2 compressors if compared to conventional 7:2 compressors which have 4 pieces of 3:2 compressors plus one final addition as shown in Figure 6. Both have 7 Exclusive-OR (XOR) gate delay. Referring to Table 4, the proposed 7:2 compressor has remarkable improvements in terms of power consumption from 0.07% (at 3.5V) to maximum of 11% (at 1.2V) and speed from 19% (at 3.5V) to 23% (at 1.2V) with respect to conventional 7:2 compressor on low voltage. Therefore, in terms of PDP, the proposed 7:2 compressor has lower value than the conventional 7:2 compressor. By utilizing HSPICE simulator, the above simulations can be carried out as expected.

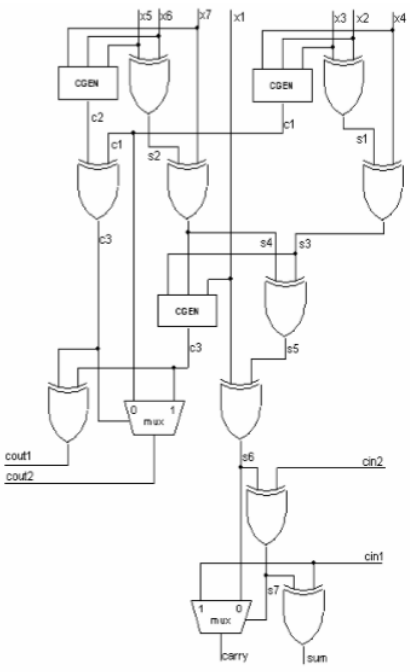


Figure 6: The Overall Proposed Structure of 7:2 Compressor [14]

Table 4: Evaluation of Proposed 7:2 Compressor with respect to Conventional 7:2 Compressor [14]

	Min. (3.5v)	Max. (1.2v)
Improvement in power consumption	0.07 %	11 %
Improvement in speed	19 %	23 %

CHAPTER 3: METHODOLOGY

3.1 RESEARCH METHODOLOGY

Figure 7: Research Methodology Used in Project

In this project, the research methodology can be categorized into four that are background study, tools used, design implementations, and further development plan.

The project started by having the background study of the concepts of compressors and different types of compressors have currently. The research papers related to compressors can be obtained from Institute of Electrical and Electronics Engineers (IEEE) Xplore Digital Library. I also approached my Final Year Project (FYP) supervisor to consult about the compressors in order to clear my doubts and have a better understanding to it.

Under tools used, it can be categorized into two, hardware and software. Under the hardware part, field-programmable gate array (FPGA) is used as it contains arrays of programmable logic blocks allowed to be configured to perform complex combination functions and it is useful when coming to designing the 8-3 compressor. Under the software part, 2 software that will be used are Altera Quartus II Web Edition and ModelSim-Altera to perform functional and timing simulation for researched compressor designs.

Under the researched designs implementation stage, the different types of researched compressor designs are implemented into FPGA to evaluate its performance in terms of fan-in, power consumption, cell area, and propagation delay. Then, the results of researched design compressors will be inter-compared with each other and determine the best way to design the 8-3 compressors which will give us the best compressor performance.

For further development, if extra time is allowed, 8-3 compressor will be designed and implemented into FPGA to evaluate and compared with the researched designs. Not to mention, the issues such as propagation delay, cell area, and power consumption have to be considered as well when designing the 8-3 compressors.

3.2 RESEARCH ACTIVITIES

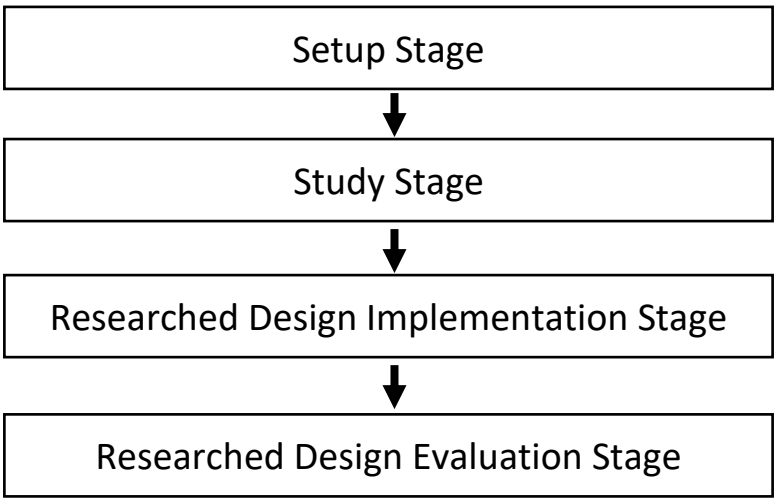


Figure 8: Research Activities Used in Project

The research activities of this project can be divided into 4 parts which are setup stage, study stage, researched design implementation stages, and researched design evaluation stage.

Under the setup stage, 2 software will be installed that are Altera Quartus II Web Edition and ModelSim-Altera to allow performing the functional and timing simulation to the compressor designed. Verilog hardware description language (HDL) will also be studied in order to have basic idea on writing the compressor design (Finite State Machine or FSM of compressor) and test bench as it is commonly used in design and verification of digital circuits at the register-transfer level (RTL).

Under the study stage, a few research papers have been studied and understood their basic idea on designing a good compressor needs. The research data has to be well-analysed in terms of propagation delay, area of cell, power consumption, and number of logic gates used.

Under the researched design implementation stage, several researched compressor designs will be implemented in into Altera Quartus II Web Edition. The researched design implemented can be known as finite-state machine. It is a model used to represent and control execution flow. The researched compressor designs are implemented using Verilog hardware description language as it includes ways of describing the propagation time and signal behaviour.

Last but not least, under the researched design evaluation stage, the researched compressor designs will be assessed by performing functional and timing simulation. Running functional simulation is to make sure that the researched compressor design implemented can handle all possible inputs and give expected output while running timing simulation is to evaluate the researched compressor design implemented from the view of real-time performance. To have better understanding on how the researched compressor designs work, it is implemented into FPGA to check its behaviour from the aspect of hardware devices. The propagation delay and power consumption will be recorded for further analysis.

3.3 PROJECT KEY MILESTONES (FINAL YEAR PROJECT I)

- ✓ Collect and Analyse Researched Design : Week 5
- ✓ Install Simulation Software : Week 6
- ✓ Study Verilog Hardware Description Language (HDL) : Week 6
- ✓ Implemented 3 Researched Designs : Week 9
- ✓ Study the Implemented Researched Designs : Week 12
- ✓ Proposed 8-3 Compressor Design Requirement : Week 13

3.4 PROJECT KEY MILESTONES (FINAL YEAR PROJECT II)

- ✓ Generate random data to be processed by compressors : Week 4
- ✓ Progress Report Submission : Week 8
- ✓ Pre-Sedex Presentation : Week 10
- ✓ Implemented 8 Researched Designs and Analyse Simulated Data : Week 11
- ✓ Draft Final Report Submission : Week 13
- ✓ Final Report & Technical Report Submission : Week 14
- ✓ Viva Presentation : Week 15

3.5 PROJECT TIMELINES (GANTT-CHART FOR FINAL YEAR PROJECT I)

Activity / Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Title Selection / Proposal														
Collect and Analyse Data														
Prepare Extended Proposal														
Install Altera Quartus II Web Edition & ModelSim-Altera and Learn to Simulate														
Implement Researched Design in Simulator														
Prepare Proposal Defense														
Study Implemented Researched Designs in Depth														
Proposed 8-3 Compressor Design Requirement														
Prepare Draft Report														
Prepare Final Report														

Table 5: Gantt chart for Final Year Project I

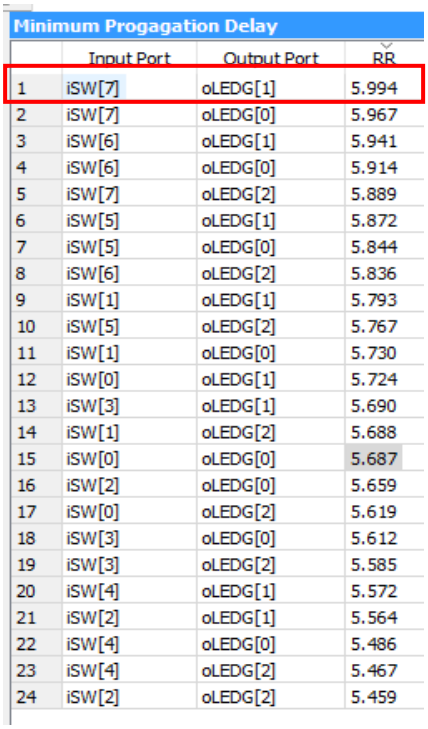
3.6 PROJECT TIMELINES (GANTT-CHART FOR FINAL YEAR PROJECT II)

Activity / Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Research on generating random data for implemented compressor designs to process														
Implement researched compressor designs														
Collect and Analyse Data														
Prepare Progress Report														
Prepare for Pre-Sedex														
Prepare Draft Final Report														
Prepare Dissertation (Soft Bound)														
Prepare Technical Paper														
Prepare Project Dissertation (Hard Bound)														

Table 6: Gantt chart for Final Year Project II

CHAPTER 4: RESULTS AND DISCUSSIONS

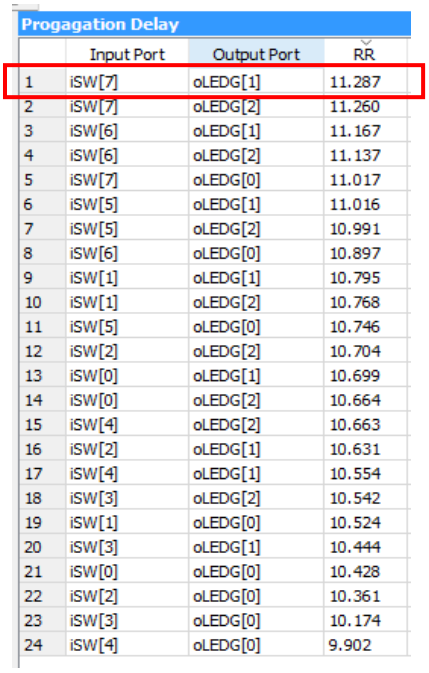
4.1 8-3 COMPRESSOR (FULL AND HALF ADDERS)



Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[1]	5.994
2	iSW[7]	oLEDG[0]	5.967
3	iSW[6]	oLEDG[1]	5.941
4	iSW[6]	oLEDG[0]	5.914
5	iSW[7]	oLEDG[2]	5.889
6	iSW[5]	oLEDG[1]	5.872
7	iSW[5]	oLEDG[0]	5.844
8	iSW[6]	oLEDG[2]	5.836
9	iSW[1]	oLEDG[1]	5.793
10	iSW[5]	oLEDG[2]	5.767
11	iSW[1]	oLEDG[0]	5.730
12	iSW[0]	oLEDG[1]	5.724
13	iSW[3]	oLEDG[1]	5.690
14	iSW[1]	oLEDG[2]	5.688
15	iSW[0]	oLEDG[0]	5.687
16	iSW[2]	oLEDG[0]	5.659
17	iSW[0]	oLEDG[2]	5.619
18	iSW[3]	oLEDG[0]	5.612
19	iSW[3]	oLEDG[2]	5.585
20	iSW[4]	oLEDG[1]	5.572
21	iSW[2]	oLEDG[1]	5.564
22	iSW[4]	oLEDG[0]	5.486
23	iSW[4]	oLEDG[2]	5.467
24	iSW[2]	oLEDG[2]	5.459

Figure 9: Minimum 8-3 Compressor Propagation Delay [12]



Progagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[1]	11.287
2	iSW[7]	oLEDG[2]	11.260
3	iSW[6]	oLEDG[1]	11.167
4	iSW[6]	oLEDG[2]	11.137
5	iSW[7]	oLEDG[0]	11.017
6	iSW[5]	oLEDG[1]	11.016
7	iSW[5]	oLEDG[2]	10.991
8	iSW[6]	oLEDG[0]	10.897
9	iSW[1]	oLEDG[1]	10.795
10	iSW[1]	oLEDG[2]	10.768
11	iSW[5]	oLEDG[0]	10.746
12	iSW[2]	oLEDG[2]	10.704
13	iSW[0]	oLEDG[1]	10.699
14	iSW[0]	oLEDG[2]	10.664
15	iSW[4]	oLEDG[2]	10.663
16	iSW[2]	oLEDG[1]	10.631
17	iSW[4]	oLEDG[1]	10.554
18	iSW[3]	oLEDG[2]	10.542
19	iSW[1]	oLEDG[0]	10.524
20	iSW[3]	oLEDG[1]	10.444
21	iSW[0]	oLEDG[0]	10.428
22	iSW[2]	oLEDG[0]	10.361
23	iSW[3]	oLEDG[0]	10.174
24	iSW[4]	oLEDG[0]	9.902

Figure 10: 8-3 Compressor Propagation Delay [12]

By referring to Figure 9 and Figure 10, it can clearly be seen that both the minimum propagation delay and simulated propagation delay has maximum value on input iSW[7] and output oLEDG[1]. This means that the critical path of 8-3 compressor model is the path that allowed signals to pass through from input iSW[7] to output oLEDG[1]. There are several values shown in Figure 9 and Figure 10. However, in this case, RR value is concerned in this design as this design is focusing on rising edge of clock at both source and destination. In order to have better understanding about the 8-3 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 11*.

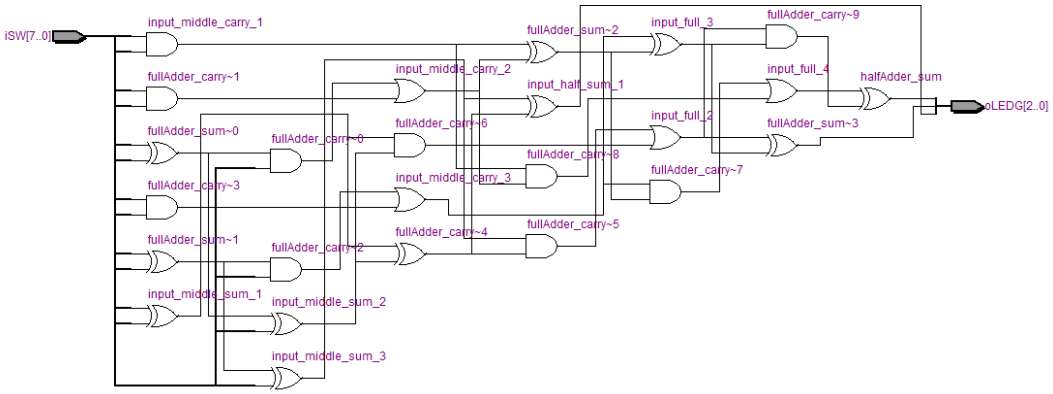


Figure 11: RTL View of 8-3 Compressor Model [12]

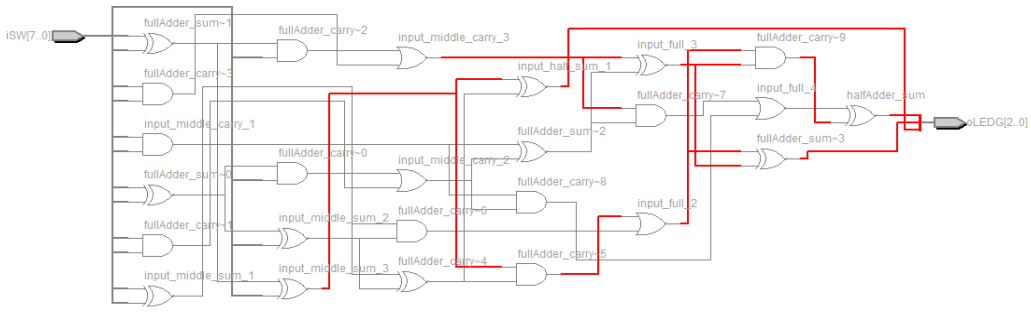


Figure 12: RTL View in Relationship with Input iSW[7] and Output oLEDG[1] [12]

Figure 11 shows the RTL view of 8-3 compressor model. As we can see, the compressor model consisted of 10 AND gates, 11 XOR gates, and 4 OR gates. With input iSW[7] and output oLEDG[1] as shown in *Figure 12*, there are 4 AND gates, 5 XOR gates, and 3 OR gates involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay.

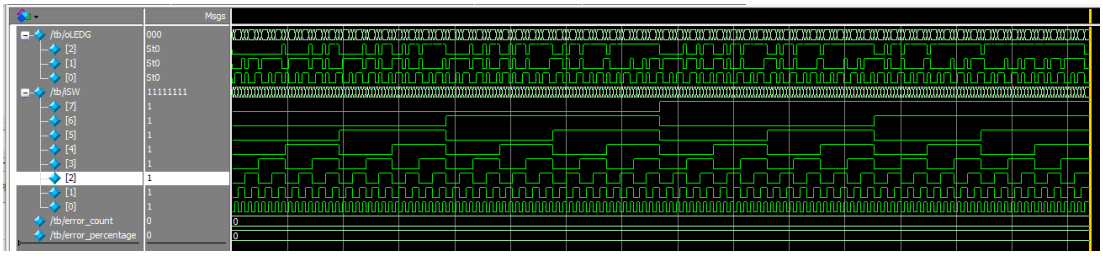


Figure 13: 8-3 Compressor Functional Simulation using ModelSim-Altera

The 8-3 compressor model and self-written functional test bench are written using Verilog Hardware Description Language as shown in *Figure 74* and *Figure 75* based on Shima et al. researched proposal [12] which consisted of 5 full adders and 2 half adders. The reason of using functional test bench simulation is to make sure that the compressor designed is functioning as expected when fixed inputs are inserted and the expected output obtained. The 8-3 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 255 (11111111_2) and observed the outputs of 8-3 compressor model ranged from 0 (000_2) to 7 (111_2). Figure 10 above shows the functional simulation for 8-3 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
The data input is 255
Total =  8, oLEDG = 0
Ok. Result = 8

Total Error Count = 0.00
The Failing Rate for Functional Testbench is 0.00
```

Figure 14: Error Result of 8-3 Compressor Functional Test bench on Console

Figure 13 above shows the number of error counts and percentage of error of outputs in 8-bits different combinations of inputs for 8-3 compressor model. The above error count is 0 after running 8-bits different combinations of inputs indicating the 8-3 compressor model is designed properly as expected.

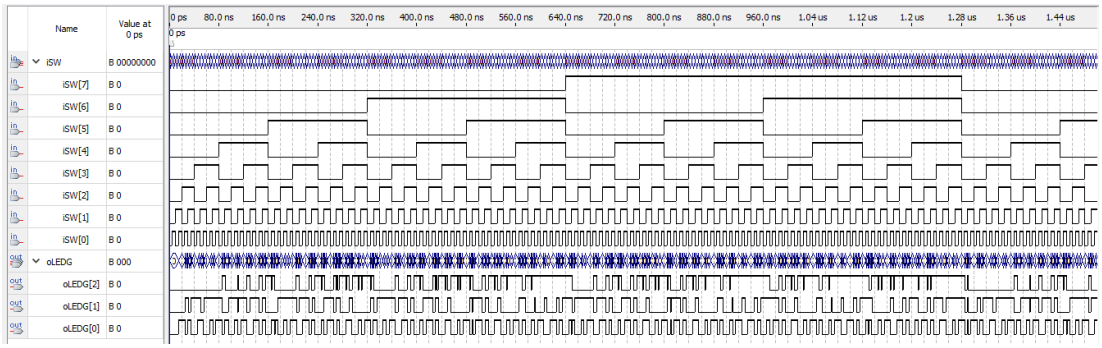


Figure 15: Overall 8-3 Compressor Model Timing Simulation

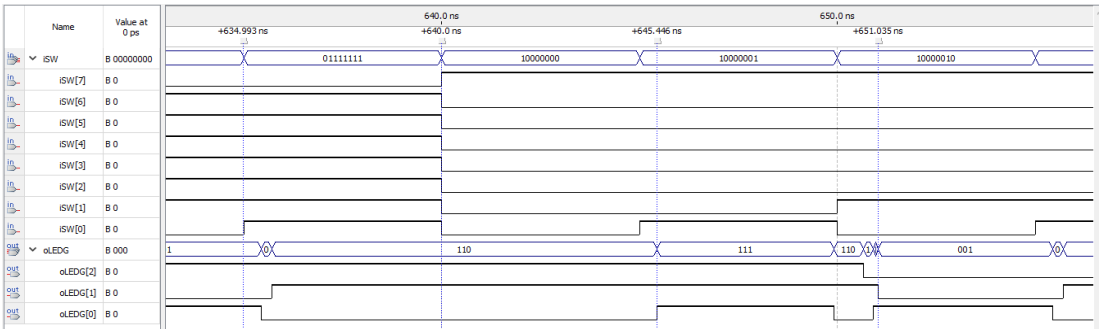


Figure 16: Timing Simulation for iSW[0..7] with bit 1s and output oLEDG[0..2]

Figure 15 shows the overall timing simulation for 8-3 compressor model designed. Timing simulation is used to emulate an implementation into real devices such as Field Programmable Gate Array (FPGA) to satisfy all the timing and functional needs. The input iSW[0] is inserted with signal 1 for each 10ns period with 50% duty cycle. The input iSW[1] is inserted with signal 1 for each 20 ns which is double period of previous iSW that is iSW[0] with also 50% duty cycle. The method of inputting the signals are applied to other inputs iSW[2..7].

Figure 16 shows the timing simulation for the critical path of 8-3 compressor model. Referring to Figure 9 and Figure 10, it can be noticed that input iSW[7] and output oLEDG[1] will have the maximum propagation delay. To prove the statement, Figure 13 data is obtained and compared in *Table 7*.

Table 7: Propagation Delay of input iSW[0..7] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Output – Input Transition**	**Percentage Error**
iSW[0..7]	634.993 ns	645.446 ns	10.453 ns	1.6462 %

4.2 8-4 COMPRESSOR (FULL AND HALF ADDERS)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[0]	oLEDG[3]	5.984
2	iSW[0]	oLEDG[1]	5.883
3	iSW[0]	oLEDG[2]	5.883
4	iSW[1]	oLEDG[3]	5.883
5	iSW[2]	oLEDG[3]	5.862
6	iSW[6]	oLEDG[3]	5.832
7	iSW[3]	oLEDG[3]	5.791
8	iSW[1]	oLEDG[1]	5.787
9	iSW[1]	oLEDG[2]	5.782
10	iSW[6]	oLEDG[1]	5.779
11	iSW[2]	oLEDG[2]	5.761
12	iSW[2]	oLEDG[1]	5.746
13	iSW[6]	oLEDG[2]	5.731
14	iSW[7]	oLEDG[3]	5.703
15	iSW[3]	oLEDG[2]	5.690
16	iSW[0]	oLEDG[0]	5.672
17	iSW[3]	oLEDG[1]	5.651
18	iSW[7]	oLEDG[1]	5.650
19	iSW[6]	oLEDG[0]	5.631
20	iSW[7]	oLEDG[2]	5.602
21	iSW[1]	oLEDG[0]	5.576
22	iSW[2]	oLEDG[0]	5.535
23	iSW[7]	oLEDG[0]	5.498
24	iSW[3]	oLEDG[0]	5.440
25	iSW[5]	oLEDG[3]	3.411
26	iSW[5]	oLEDG[1]	3.358
27	iSW[5]	oLEDG[2]	3.310
28	iSW[5]	oLEDG[0]	3.207
29	iSW[4]	oLEDG[3]	3.188
30	iSW[4]	oLEDG[2]	3.106
31	iSW[4]	oLEDG[1]	3.005
32	iSW[4]	oLEDG[0]	2.793

Figure 17: Minimum 8-4 Compressor Model Propagation Delay [1]

Progagation Delay

	Input Port	Output Port	RR
1	iSW[0]	oLEDG[3]	11.305
2	iSW[6]	oLEDG[3]	11.217
3	iSW[0]	oLEDG[2]	11.109
4	iSW[1]	oLEDG[3]	11.035
5	iSW[6]	oLEDG[2]	11.021
6	iSW[2]	oLEDG[3]	10.992
7	iSW[0]	oLEDG[1]	10.973
8	iSW[7]	oLEDG[3]	10.909
9	iSW[1]	oLEDG[2]	10.839
10	iSW[6]	oLEDG[1]	10.825
11	iSW[2]	oLEDG[2]	10.796
12	iSW[3]	oLEDG[3]	10.784
13	iSW[1]	oLEDG[1]	10.730
14	iSW[2]	oLEDG[1]	10.726
15	iSW[7]	oLEDG[2]	10.713
16	iSW[3]	oLEDG[2]	10.588
17	iSW[7]	oLEDG[1]	10.517
18	iSW[3]	oLEDG[1]	10.512
19	iSW[0]	oLEDG[0]	10.380
20	iSW[6]	oLEDG[0]	10.291
21	iSW[1]	oLEDG[0]	10.110
22	iSW[2]	oLEDG[0]	10.067
23	iSW[7]	oLEDG[0]	9.983
24	iSW[3]	oLEDG[0]	9.859
25	iSW[5]	oLEDG[3]	7.276
26	iSW[5]	oLEDG[2]	7.080
27	iSW[4]	oLEDG[3]	6.966
28	iSW[5]	oLEDG[1]	6.884
29	iSW[4]	oLEDG[2]	6.770
30	iSW[4]	oLEDG[1]	6.749
31	iSW[5]	oLEDG[0]	6.350
32	iSW[4]	oLEDG[0]	5.370

Figure 18: 8-4 Compressor Model Maximum Propagation Delay [1]

By referring to *Figure 17* and *Figure 18*, it can clearly be seen that both the minimum propagation delay and simulated propagation delay has maximum value on input iSW[0] and output oLEDG[3]. This means that the critical path of 8-4 compressor model is the path that allowed signals to pass through from input iSW[0] to output oLEDG[3]. There are several values shown in *Figure 17* and *Figure 18*. However, in this case, RR value is concerned in this design as this design is focusing on rising edge of clock at both source and destination. The critical path is the path between the input and output with maximum circuit delay. To have better understanding about 8-4 compressor model, RTL view of circuit is shown in *Figure 19*.

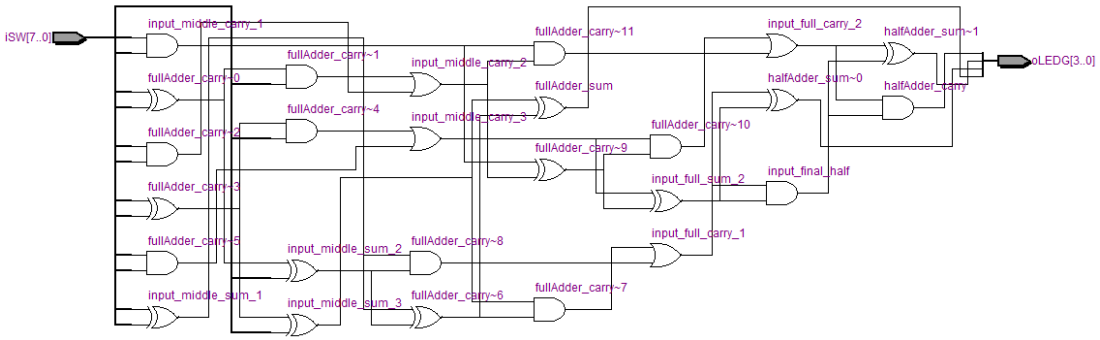


Figure 19: Overall RTL View of 8-4 Compressor Model [1]

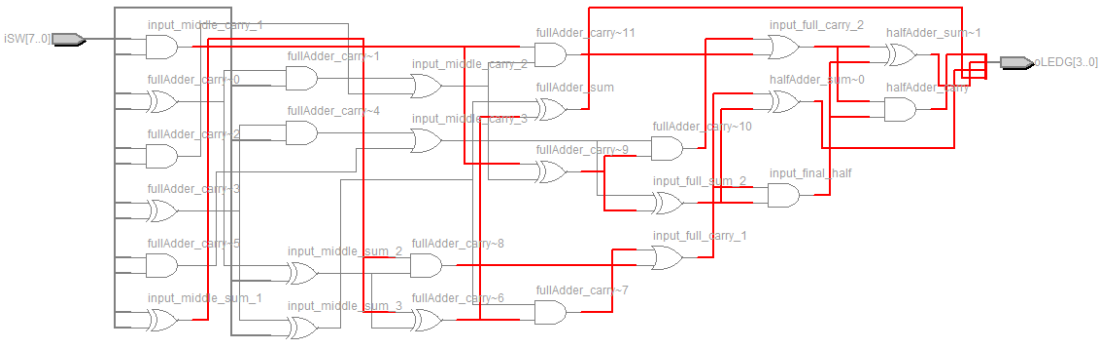


Figure 20: RTL View in Relationship with Input iSW[0] and Output oLEDG[3] [1]

Figure 19 shows the RTL view of 8-4 compressor model. As we can see, the compressor model consisted of 11 AND gates, 11 XOR gates, and 2 OR gates. With input iSW[0] and output oLEDG[3] as shown in *Figure 20*, there are 7 AND gates, 7 XOR gates, and 2 OR gates involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, when the number of XOR gates increases, the propagation delay of the compressor will be increased dramatically.

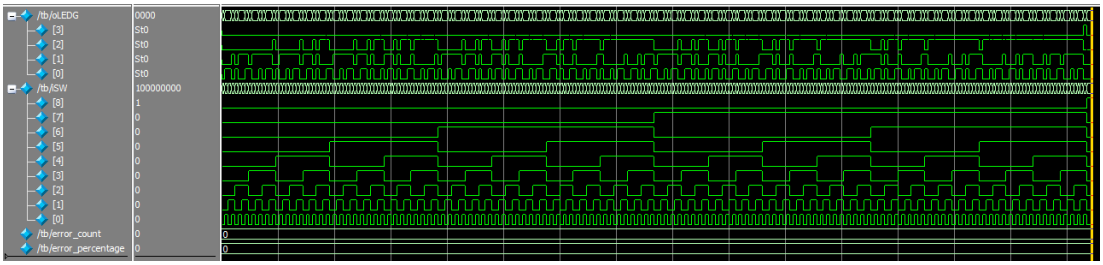


Figure 21: 8-4 Compressor Functional Simulation using ModelSim-Altera

The 8-4 compressor model and self-written functional test bench are written using Verilog Hardware Description Language as shown in *Figure 76* and *Figure 77* based on R.Marimuthu, et al. researched proposal [1] which consisted of 4 full adders and 3 half adders. The 8-4 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 255 (11111111_2) and observed the outputs of 8-4 compressor model ranged from 0 (0000_2) to 8 (1000_2). *Figure 21* above shows the functional simulation for 8-4 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
The data input is 255
Total =  8, oLEDG =  8
Ok. Result = 8

Total Error Count = 0.00
The Failing Rate for Functional Testbench is 0.00
```

Figure 22: Error Result of 8-4 Compressor Functional Test bench on Console

Figure 22 above shows the number of error counts and percentage of error of outputs in 8-bits different combinations of inputs for 8-4 compressor model. The above error count is 0 after running 8-bits different combinations of inputs indicating the 8-4 compressor model is designed properly as expected.

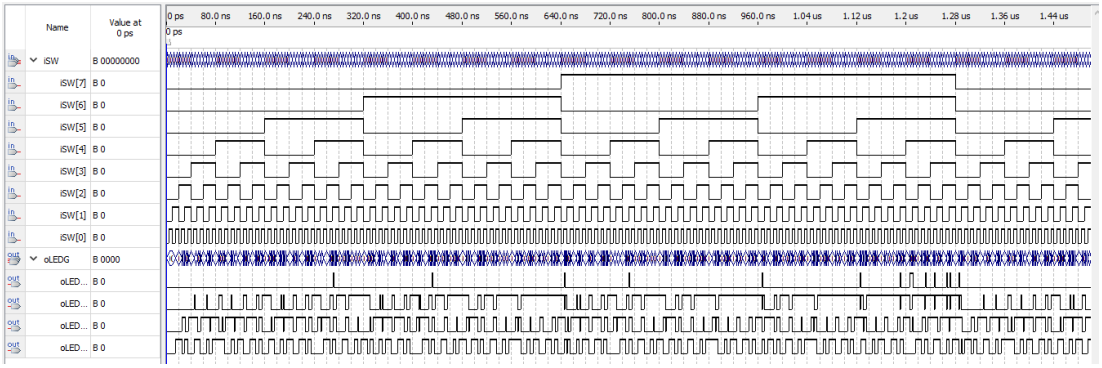


Figure 23: Overall 8-4 Compressor Model Timing Simulation

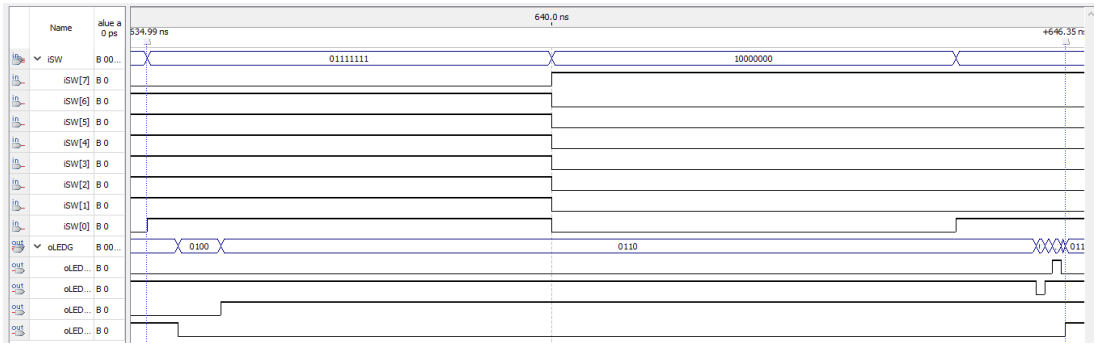


Figure 24: Timing Simulation for Inputs iSW[0..7] with bit 1 and output oLEDG[0..3]

Figure 23 shows the overall timing simulation for 8-4 compressor model designed. The input iSW[0] is inserted with signal 1 for each 10ns period with 50% duty cycle. The input iSW[1] is inserted with signal 1 for each 20 ns which is double period of previous iSW that is iSW[0] with also 50% duty cycle. The method of inputting the signals are applied to other inputs iSW[2..7].

Figure 24 shows the timing simulation for the critical path of 8-4 compressor model. Referring to *Figure 17* and *Figure 18*, it can be noticed that input iSW[0] and output oLEDG[2] will have the maximum propagation delay. To prove the statement, *Figure 24* data is obtained and compared in *Table 8*.

Table 8: Propagation Delay of iSW[0..7] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Output – Input Transition**	**Percentage Error**
iSW[0..7]	634.99 ns	646.35 ns	11.36 ns	1.789 %

4.3 9-4 COMPRESSOR (FULL AND HALF ADDERS)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[3]	6.211
2	iSW[6]	oLEDG[3]	6.150
3	iSW[8]	oLEDG[3]	6.095
4	iSW[7]	oLEDG[1]	6.053
5	iSW[2]	oLEDG[3]	6.004
6	iSW[6]	oLEDG[1]	5.991
7	iSW[7]	oLEDG[0]	5.968
8	iSW[4]	oLEDG[3]	5.965
9	iSW[1]	oLEDG[3]	5.962
10	iSW[7]	oLEDG[2]	5.947
11	iSW[8]	oLEDG[1]	5.940
12	iSW[6]	oLEDG[0]	5.906
13	iSW[6]	oLEDG[2]	5.886
14	iSW[3]	oLEDG[3]	5.871
15	iSW[8]	oLEDG[0]	5.855
16	iSW[4]	oLEDG[1]	5.841
17	iSW[5]	oLEDG[3]	5.838
18	iSW[8]	oLEDG[2]	5.831
19	iSW[0]	oLEDG[3]	5.802
20	iSW[2]	oLEDG[1]	5.802
21	iSW[3]	oLEDG[1]	5.799
22	iSW[1]	oLEDG[1]	5.760
23	iSW[4]	oLEDG[0]	5.752
24	iSW[2]	oLEDG[2]	5.747
25	iSW[3]	oLEDG[0]	5.742
26	iSW[2]	oLEDG[0]	5.710
27	iSW[1]	oLEDG[2]	5.707
28	iSW[4]	oLEDG[2]	5.701
29	iSW[1]	oLEDG[0]	5.668
30	iSW[5]	oLEDG[1]	5.636
31	iSW[3]	oLEDG[2]	5.607
32	iSW[0]	oLEDG[1]	5.600
33	iSW[5]	oLEDG[2]	5.595
34	iSW[5]	oLEDG[0]	5.556
35	iSW[0]	oLEDG[2]	5.545
36	iSW[0]	oLEDG[0]	5.508

Figure 25: Minimum 9-4 Compressor Model Propagation Delay [1]

Progagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[3]	11.731
2	iSW[5]	oLEDG[3]	11.615
3	iSW[6]	oLEDG[3]	11.597
4	iSW[7]	oLEDG[1]	11.474
5	iSW[8]	oLEDG[3]	11.459
6	iSW[5]	oLEDG[1]	11.358
7	iSW[6]	oLEDG[1]	11.340
8	iSW[2]	oLEDG[3]	11.334
9	iSW[7]	oLEDG[2]	11.284
10	iSW[1]	oLEDG[3]	11.227
11	iSW[4]	oLEDG[3]	11.225
12	iSW[8]	oLEDG[1]	11.202
13	iSW[5]	oLEDG[2]	11.168
14	iSW[6]	oLEDG[2]	11.150
15	iSW[3]	oLEDG[3]	11.144
16	iSW[2]	oLEDG[1]	11.077
17	iSW[7]	oLEDG[0]	11.020
18	iSW[8]	oLEDG[2]	11.012
19	iSW[1]	oLEDG[1]	10.970
20	iSW[6]	oLEDG[0]	10.889
21	iSW[2]	oLEDG[2]	10.887
22	iSW[4]	oLEDG[1]	10.874
23	iSW[0]	oLEDG[3]	10.845
24	iSW[3]	oLEDG[1]	10.799
25	iSW[1]	oLEDG[2]	10.780
26	iSW[4]	oLEDG[2]	10.778
27	iSW[8]	oLEDG[0]	10.750
28	iSW[3]	oLEDG[2]	10.697
29	iSW[0]	oLEDG[1]	10.588
30	iSW[4]	oLEDG[0]	10.549
31	iSW[2]	oLEDG[0]	10.497
32	iSW[3]	oLEDG[0]	10.468
33	iSW[0]	oLEDG[2]	10.398
34	iSW[1]	oLEDG[0]	10.392
35	iSW[5]	oLEDG[0]	10.052
36	iSW[0]	oLEDG[0]	10.007

Figure 26: 9-4 Compressor Model Maximum Propagation Delay [1]

By referring to *Figure 25* and *Figure 26*, it can clearly be seen that both the minimum propagation delay and simulated propagation delay has maximum value on input iSW[7] and output oLEDG[3]. This means that the critical path of 9-4 compressor model is the path that allowed signals to pass through from input iSW[7] to output oLEDG[3]. The critical path is the path between the input and output with maximum circuit delay. There are several values shown in *Figure 25* and *Figure 26*. However, in this case, RR value is concerned in this design as this design is focusing on rising edge of clock at both source and destination. In order to have better understanding about the 9-4 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 27*.

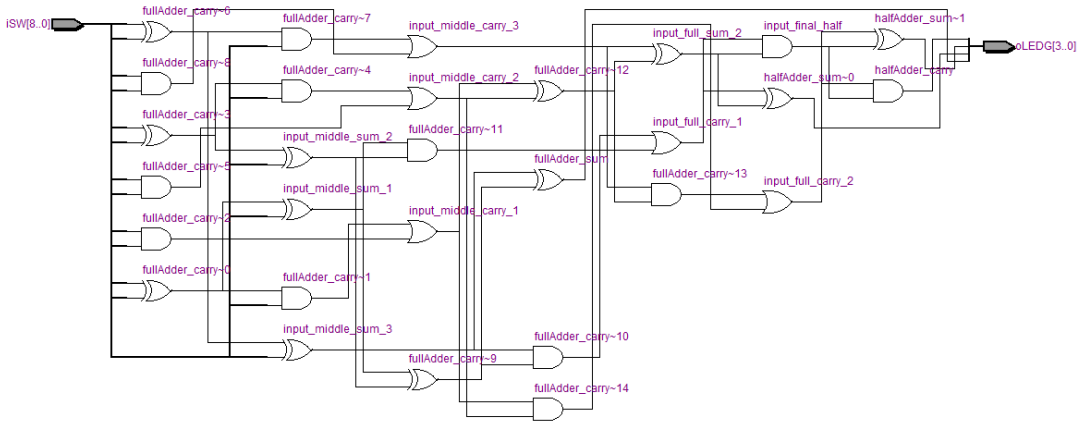


Figure 27: Overall RTL View of 9-4 Compressor Model [1]

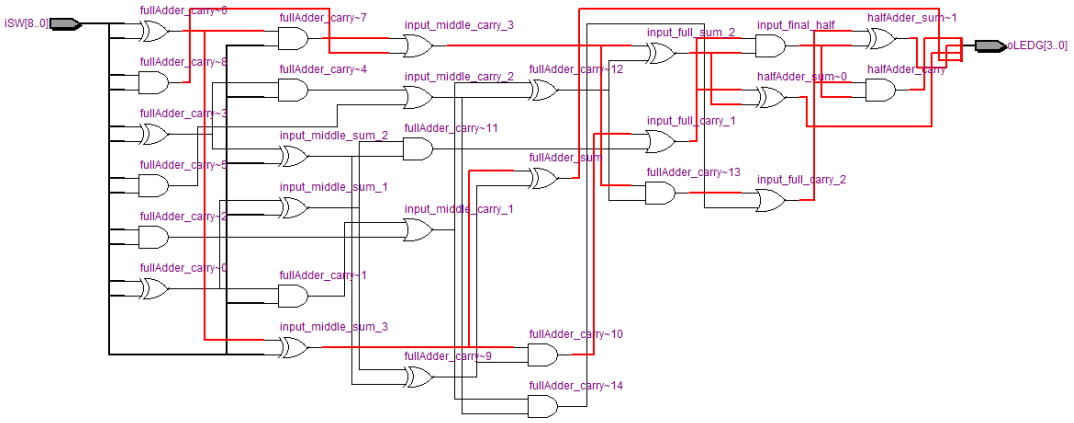


Figure 28: RTL View in Relationship with Input iSW[7] and Output oLEDG[3] [1]

Figure 27 shows the RTL view of 9-4 compressor model. As we can see, the compressor model consisted of 12 AND gates, 12 XOR gates, and 5 OR gates. With input iSW[7] and output oLEDG[3] as shown in *Figure 28*, there are 6 AND gates, 6 XOR gates, and 3 OR gates involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, when the number of XOR gates increases, the propagation delay of the compressor will be increased dramatically.

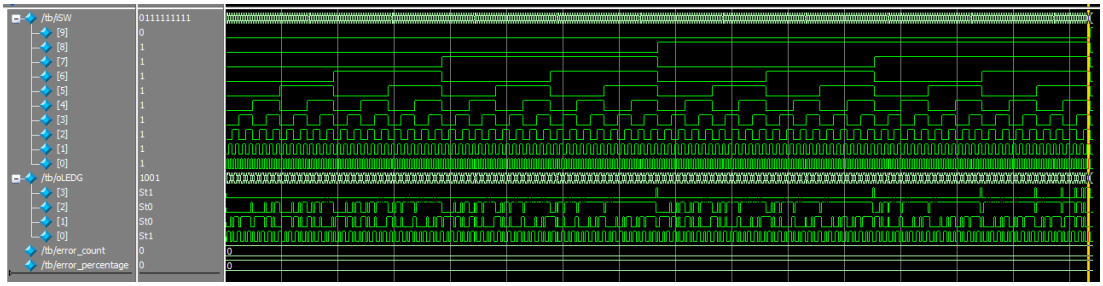


Figure 29: 9-4 Compressor Functional Simulation using ModelSim-Altera

The 9-4 compressor model and self-written functional test bench are written using Verilog Hardware Description Language as shown in *Figure 78* and *Figure 79* based on R.Marimuthu, et al. researched proposal [1] which consisted of 5 full adders and 2 half adders. The 9-4 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 511 (111111111_2) and observed the outputs of 9-4 compressor model ranged from 0 (0000_2) to 9 (1001_2). *Figure 29* above shows the functional simulation for 9-4 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
The data input is  511
Total =  9, oLEDG =  9
Ok. Result = 9

Total Error Count = 0.00
The Failing Rate for Functional Testbench is 0.00
```

Figure 30: Error Result of 9-4 Compressor Functional Test bench on Console

Figure 30 above shows the number of error counts and percentage of error of outputs in 9-bits different combinations of inputs for 9-4 compressor model. The above error count is 0 after running 9-bits different combinations of inputs indicating the 9-4 compressor model is designed properly as expected.

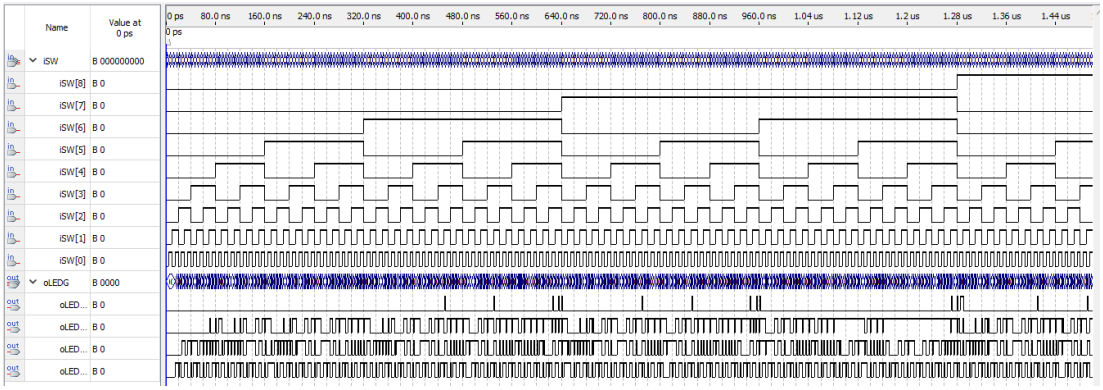


Figure 31: Overall 9-4 Compressor Model Timing Simulation

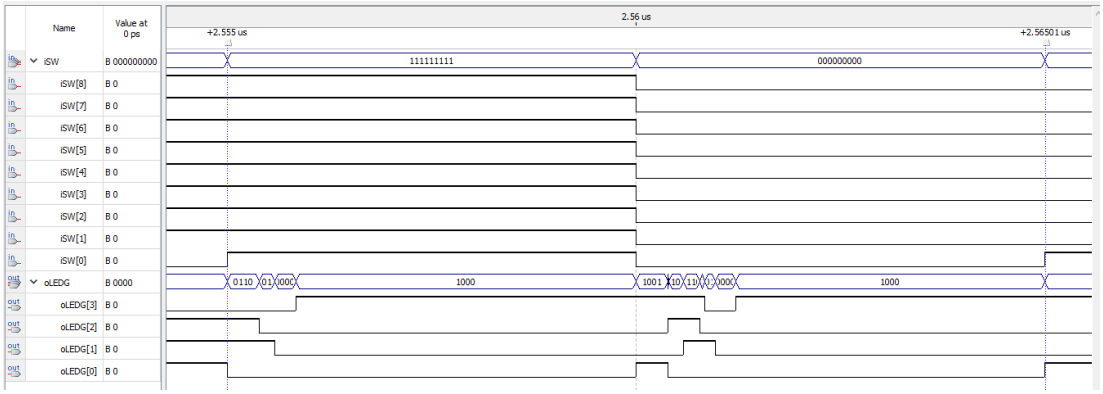


Figure 32: Timing Simulation for Inputs iSW[0..8] with bit 1s and output oLEDG[0..3]

Figure 31 shows the overall timing simulation for 9-4 compressor model designed. The input iSW[0] is inserted with signal 1 for each 10ns period with 50% duty cycle. The input iSW[1] is inserted with signal 1 for each 20 ns which is double period of previous iSW that is iSW[0] with also 50% duty cycle. The method of inputting the signals are applied to other inputs iSW[2..7].

Figure 32 shows the timing simulation for the critical path of 9-4 compressor model. Referring to *Figure 25* and *Figure 26*, it can be noticed that input iSW[7] and output oLEDG[3] will have the maximum propagation delay. To prove the statement, *Figure 32* data is obtained and compared in *Table 9*.

Table 9: Propagation Delay of iSW[0..8] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Output – Input Transition**	**Percentage Error**
iSW[0..8]	2555 ns	2565.01 us	10.01 ns	0.3918 %

4.4 9-3 COMPRESSOR (FULL AND HALF ADDERS)

Minimum Propagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[0]	5.958
2	iSW[7]	oLEDG[1]	5.926
3	iSW[6]	oLEDG[0]	5.896
4	iSW[6]	oLEDG[1]	5.865
5	iSW[8]	oLEDG[0]	5.845
6	iSW[4]	oLEDG[1]	5.837
7	iSW[7]	oLEDG[2]	5.813
8	iSW[8]	oLEDG[1]	5.810
9	iSW[3]	oLEDG[1]	5.779
10	iSW[4]	oLEDG[0]	5.757
11	iSW[6]	oLEDG[2]	5.752
12	iSW[3]	oLEDG[0]	5.747
13	iSW[4]	oLEDG[2]	5.724
14	iSW[2]	oLEDG[0]	5.700
15	iSW[8]	oLEDG[2]	5.697
16	iSW[2]	oLEDG[1]	5.693
17	iSW[3]	oLEDG[2]	5.666
18	iSW[1]	oLEDG[0]	5.661
19	iSW[1]	oLEDG[1]	5.650
20	iSW[5]	oLEDG[1]	5.643
21	iSW[2]	oLEDG[2]	5.580
22	iSW[5]	oLEDG[0]	5.556
23	iSW[5]	oLEDG[2]	5.540
24	iSW[1]	oLEDG[2]	5.537
25	iSW[0]	oLEDG[0]	5.494
26	iSW[0]	oLEDG[1]	5.490
27	iSW[0]	oLEDG[2]	5.377

Figure 33: Minimum 9-3 Compressor Model Propagation Delay [12]

Propagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[1]	6.045
2	iSW[5]	oLEDG[2]	6.037
3	iSW[5]	oLEDG[1]	6.025
4	iSW[6]	oLEDG[1]	5.983
5	iSW[7]	oLEDG[0]	5.958
6	iSW[7]	oLEDG[2]	5.942
7	iSW[8]	oLEDG[1]	5.932
8	iSW[6]	oLEDG[0]	5.896
9	iSW[6]	oLEDG[2]	5.880
10	iSW[4]	oLEDG[2]	5.849
11	iSW[8]	oLEDG[0]	5.845
12	iSW[4]	oLEDG[1]	5.844
13	iSW[3]	oLEDG[1]	5.834
14	iSW[8]	oLEDG[2]	5.829
15	iSW[3]	oLEDG[2]	5.812
16	iSW[2]	oLEDG[1]	5.787
17	iSW[4]	oLEDG[0]	5.757
18	iSW[1]	oLEDG[1]	5.748
19	iSW[3]	oLEDG[0]	5.747
20	iSW[2]	oLEDG[2]	5.704
21	iSW[2]	oLEDG[0]	5.700
22	iSW[1]	oLEDG[0]	5.661
23	iSW[1]	oLEDG[2]	5.661
24	iSW[0]	oLEDG[1]	5.581
25	iSW[5]	oLEDG[0]	5.556
26	iSW[0]	oLEDG[2]	5.501
27	iSW[0]	oLEDG[0]	5.494

Figure 34: 9-3 Compressor Model Maximum Propagation Delay [12]

By referring to *Figure 33* and *Figure 34*, it can clearly be seen that the minimum propagation delay and simulated propagation delay has maximum value on input iSW[7] and output oLEDG[0] and maximum value on input iSW[7] and output oLEDG[1] respectively. In this case, the maximum propagation delay on *Figure 34* is considered which the critical path of 9-3 compressor model is the path that allowed signals to pass through from input iSW[7] to output oLEDG[1]. To have better understanding to the 9-3 compressor model, RTL view of circuit is shown in *Figure 35*.

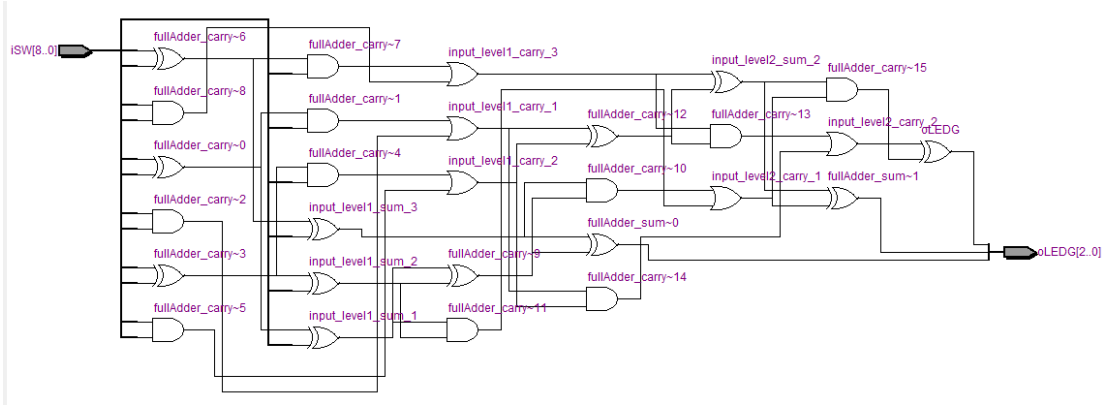


Figure 35: Overall RTL View of 9-3 Compressor Model [12]

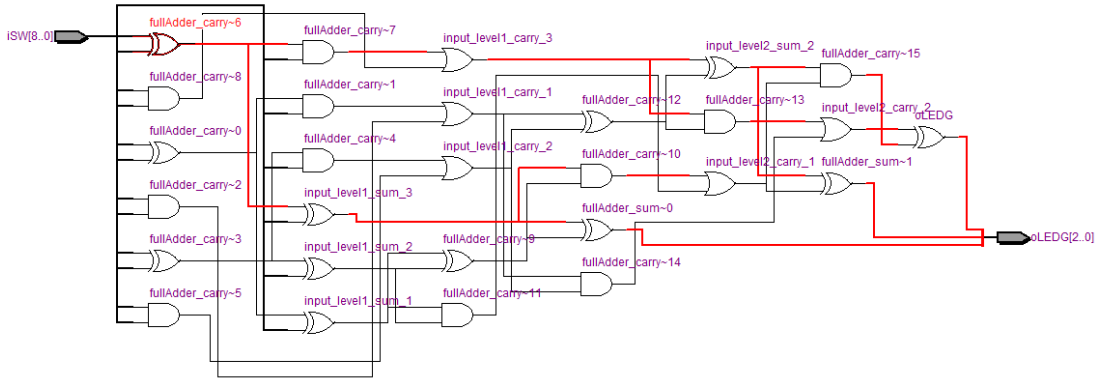


Figure 36: RTL View in Relationship with Input iSW[7] and Output oLEDG[1] [12]

Figure 35 shows the RTL view of 9-3 compressor model. As we can see, the compressor model consisted of 11 AND gates, 12 XOR gates, and 5 OR gates. With input iSW[7] and output oLEDG[1] as shown in *Figure 36*, there are 4 AND gates, 6 XOR gates, and 3 OR gates involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, when the number of XOR gates increases, the propagation delay of the compressor will be increased dramatically.

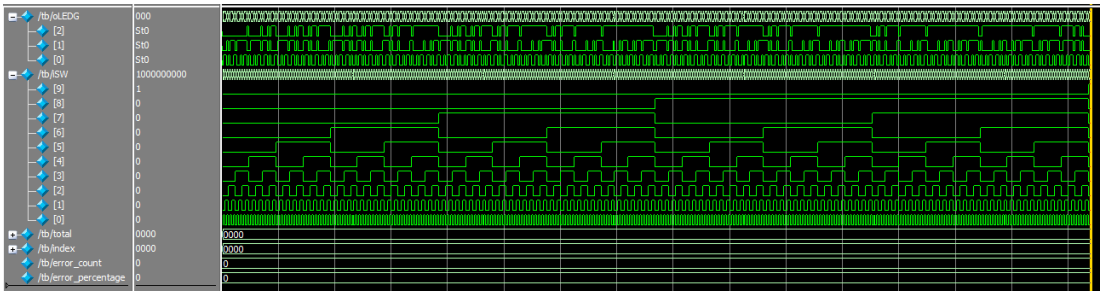


Figure 37: 9-3 Compressor Functional Simulation using ModelSim-Altera

The 9-3 compressor model and self-written functional test bench are written using Verilog Hardware Description Language (HDL) as shown in *Figure 80* and *Figure 81* based on Shima Mehrabi, et al. researched proposal [12] which consisted of 6 full adders and 1 half adders. The 9-3 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 511 (111111111_2) and observed the outputs of 9-4 compressor model ranged from 0 (0000_2) to 9 (1001_2). *Figure 38* above shows the functional simulation for 9-3 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
# The data input is  510
# Total =  8, oLEDG = 0
# Ok. Result = 0
#
# The data input is  511
# Total =  9, oLEDG = 1
# Ok. Result = 1
#
# Total Error Count = 0.00
# The Failing Rate for Functional Testbench is 0.00
```

Figure 38: Error Result of 9-3 Compressor Functional Test bench on Console

Figure 38 above shows the number of error counts and percentage of error of outputs in 9-bits different combinations of inputs for 9-3 compressor model. The above error count is 0 after running 9-bits different combinations of inputs indicating the 9-3 compressor model is designed properly as expected.

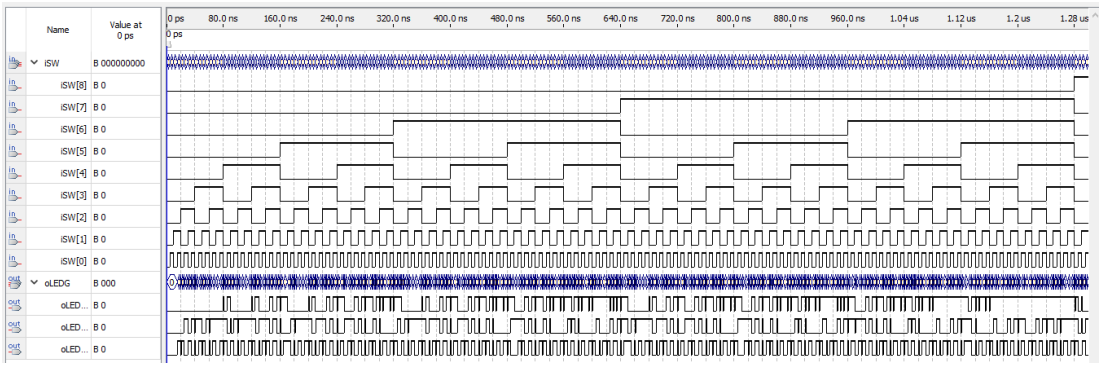


Figure 39: Overall 9-3 Compressor Model Timing Simulation

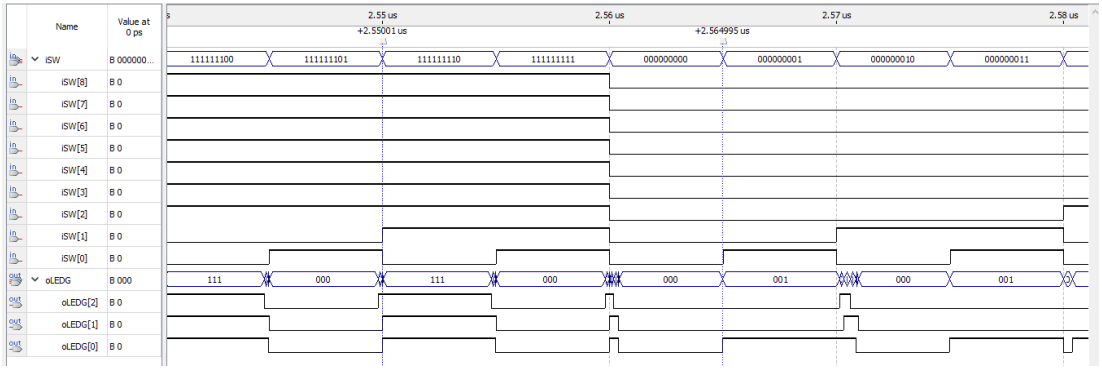


Figure 40: Timing Simulation for Inputs iSW[0..8] with bit 1s and output oLEDG[0..3]

Figure 39 shows the overall timing simulation for 9-3 compressor model designed. The method of inserting signal for timing simulation is same as previously discussed 9-4 Compressor using Adder Method. *Figure 40* shows the timing simulation for the critical path of 9-3 compressor model. Referring to Figure 33, it can be noticed that input iSW[7] and output oLEDG[1] will have the maximum propagation delay. To prove the statement, Figure 39 data is obtained and compared in Table 8.

Table 10: Propagation Delay of iSW[0..8] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Time Between Obtained**	**Percentage Error**
iSW[0..8]	2.55001 us	2.564995 us	14.985 ns	0.59 %

4.5 7-3 COMPRESSOR (FULL AND HALF ADDERS)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[0]	5.973
2	iSW[6]	oLEDG[0]	5.908
3	iSW[4]	oLEDG[0]	5.844
4	iSW[1]	oLEDG[0]	5.782
5	iSW[2]	oLEDG[0]	5.686
6	iSW[5]	oLEDG[1]	5.679
7	iSW[2]	oLEDG[1]	5.633
8	iSW[6]	oLEDG[1]	5.613
9	iSW[1]	oLEDG[1]	5.593
10	iSW[5]	oLEDG[2]	5.589
11	iSW[4]	oLEDG[1]	5.552
12	iSW[2]	oLEDG[2]	5.546
13	iSW[6]	oLEDG[2]	5.523
14	iSW[1]	oLEDG[2]	5.506
15	iSW[3]	oLEDG[1]	5.503
16	iSW[3]	oLEDG[0]	5.476
17	iSW[4]	oLEDG[2]	5.462
18	iSW[3]	oLEDG[2]	5.422
19	iSW[0]	oLEDG[1]	5.359
20	iSW[0]	oLEDG[0]	5.332
21	iSW[0]	oLEDG[2]	5.278

Figure 41: Minimum 7-3 Compressor Model Minimum Propagation Delay [10]

Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[1]	11.204
2	iSW[5]	oLEDG[2]	11.066
3	iSW[5]	oLEDG[0]	11.050
4	iSW[6]	oLEDG[1]	11.039
5	iSW[6]	oLEDG[2]	10.901
6	iSW[4]	oLEDG[1]	10.893
7	iSW[6]	oLEDG[0]	10.885
8	iSW[1]	oLEDG[1]	10.784
9	iSW[4]	oLEDG[2]	10.755
10	iSW[4]	oLEDG[0]	10.739
11	iSW[1]	oLEDG[2]	10.646
12	iSW[1]	oLEDG[0]	10.632
13	iSW[2]	oLEDG[1]	10.566
14	iSW[2]	oLEDG[2]	10.428
15	iSW[2]	oLEDG[0]	10.414
16	iSW[3]	oLEDG[1]	10.179
17	iSW[3]	oLEDG[2]	10.039
18	iSW[3]	oLEDG[0]	9.836
19	iSW[0]	oLEDG[1]	9.755
20	iSW[0]	oLEDG[2]	9.617
21	iSW[0]	oLEDG[0]	9.602

Figure 42: 7-3 Compressor Model Maximum Propagation Delay [10]

By referring to Figure 41 and *Figure 42*, it can clearly be seen that the minimum propagation delay and simulated propagation delay has maximum value on input iSW[5] and output oLEDG[0] and maximum value on input iSW[5] and output oLEDG[1] respectively. In this case, the maximum propagation delay on *Figure 42* will be considered which the critical path of 7-3 compressor model is the path that allowed signals to pass through from input iSW[5] to output oLEDG[1]. The critical path is the path between the input and output with maximum circuit delay. In order to have better understanding about the 7-3 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 43*.

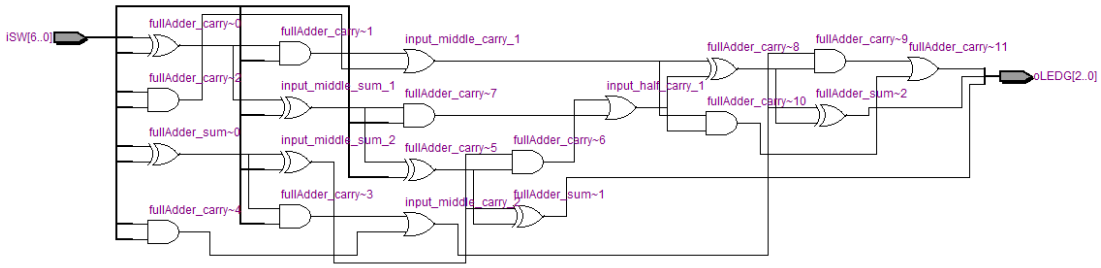


Figure 43: Overall RTL View of 7-3 Compressor Model [10]

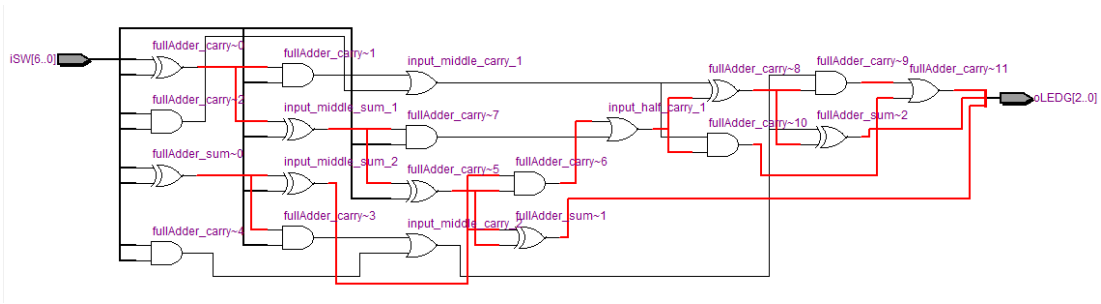


Figure 44: RTL View in Relationship with Input iSW[5] and Output oLEDG[1] [10]

Figure 43 shows the RTL view of 7-3 compressor model. As we can see, the compressor model consisted of 8 AND gates, 8 XOR gates, and 4 OR gates. With input iSW[5] and output oLEDG[1] as shown in *Figure 44*, there are 2 AND gates, 7 XOR gates, and 1 OR gate involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, when the number of XOR gates increases, the propagation delay of the compressor will be increased dramatically.

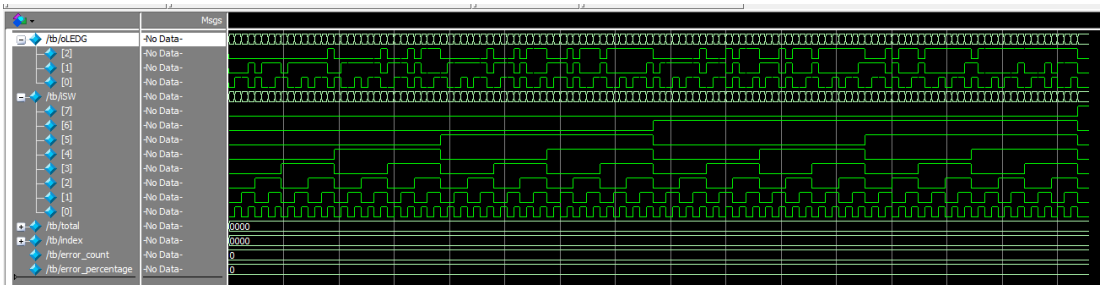


Figure 45: 7-3 Compressor Functional Simulation using ModelSim-Altera

The 7-3 compressor model and self-written functional test bench are written using Verilog Hardware Description Language (HDL) as shown in Figure 82 and Figure 83 based on Shima Mehrabi, et al. researched proposal [12] which consisted of 4 full adders. The 7-3 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 128 (1111111_2) and observed the outputs of 7-3 compressor model ranged from 0 (000_2) to 7 (111_2). *Figure 46* above shows the functional simulation for 7-3 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
# The data input is 126
# Total =  6, oLEDG = 6
# Ok. Result = 6
#
# The data input is 127
# Total =  7, oLEDG = 7
# Ok. Result = 7
#
# Total Error Count = 0.00
# The Failing Rate for Functional Testbench is 0.00
#
```

Figure 46: Error Result of 7-3 Compressor Functional Test bench on Console

Figure 46 above shows the number of error counts and percentage of error of outputs in 7-bits different combinations of inputs for 7-3 compressor model. The above error count is 0 after running 7-bits different combinations of inputs indicating the 7-3 compressor model is designed properly as expected.

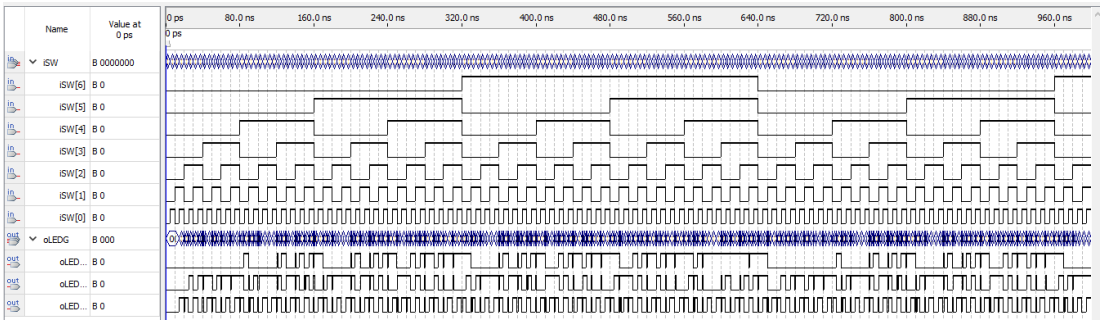


Figure 47: Overall 7-3 Compressor Model Timing Simulation

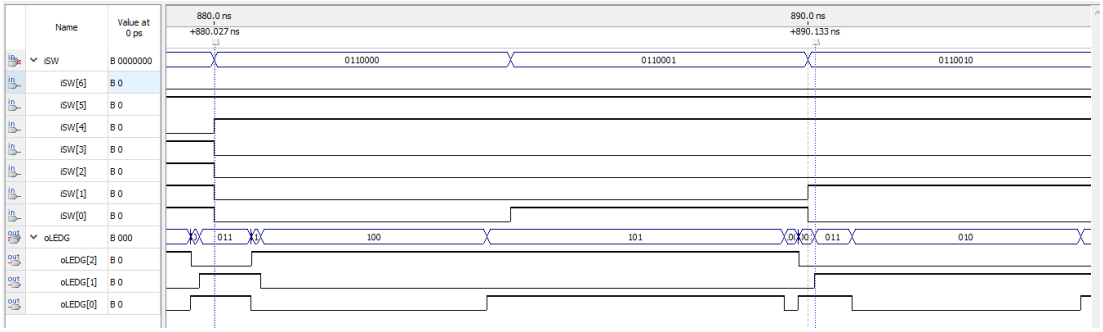


Figure 48: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[1] with bit 1

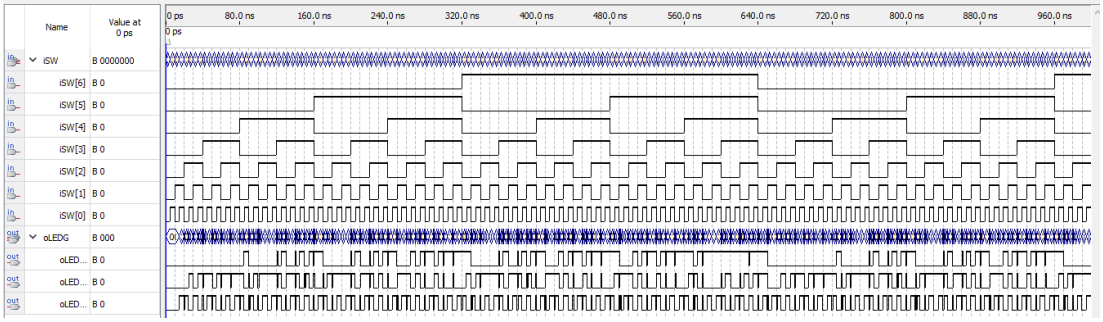


Figure 47 shows the overall timing simulation for 7-3 compressor model designed. The method of inserting signal for timing simulation is same as previously discussed 7-3 Compressor using Adder Method. *Figure 48* shows the timing simulation for the critical path of 7-3 compressor model. Referring to *Figure 42*, it can be noticed that input iSW[5] and output oLEDG[1] will have the maximum propagation delay. To prove the statement, Figure 39 data is obtained and compared in Table 8.

Table 11: Propagation Delay of iSW[5] and oLEDG[1] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Output – Input Transition**	**Percentage Error**
iSW[5]	880.027 ns	890.133 ns	14.985 ns	1.1484 %

4.6 7-3 COMPRESSOR (MUX AND XOR LOGIC GATES)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[0]	5.977
2	iSW[6]	oLEDG[0]	5.908
3	iSW[4]	oLEDG[0]	5.844
4	iSW[2]	oLEDG[0]	5.814
5	iSW[3]	oLEDG[0]	5.780
6	iSW[5]	oLEDG[1]	5.682
7	iSW[2]	oLEDG[1]	5.633
8	iSW[6]	oLEDG[1]	5.613
9	iSW[5]	oLEDG[2]	5.592
10	iSW[3]	oLEDG[1]	5.570
11	iSW[4]	oLEDG[1]	5.552
12	iSW[2]	oLEDG[2]	5.546
13	iSW[6]	oLEDG[2]	5.523
14	iSW[1]	oLEDG[1]	5.507
15	iSW[1]	oLEDG[0]	5.482
16	iSW[3]	oLEDG[2]	5.480
17	iSW[4]	oLEDG[2]	5.462
18	iSW[1]	oLEDG[2]	5.426
19	iSW[0]	oLEDG[1]	5.326
20	iSW[0]	oLEDG[0]	5.300
21	iSW[0]	oLEDG[2]	5.245

Figure 49: Minimum 7-3 Compressor Model Minimum Propagation Delay using MUX and XOR Logic Gates [10]

Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[1]	11.210
2	iSW[5]	oLEDG[2]	11.072
3	iSW[5]	oLEDG[0]	11.056
4	iSW[6]	oLEDG[1]	11.039
5	iSW[6]	oLEDG[2]	10.901
6	iSW[4]	oLEDG[1]	10.893
7	iSW[6]	oLEDG[0]	10.885
8	iSW[2]	oLEDG[1]	10.872
9	iSW[4]	oLEDG[2]	10.755
10	iSW[4]	oLEDG[0]	10.739
11	iSW[2]	oLEDG[2]	10.734
12	iSW[2]	oLEDG[0]	10.722
13	iSW[3]	oLEDG[1]	10.706
14	iSW[3]	oLEDG[2]	10.568
15	iSW[3]	oLEDG[0]	10.556
16	iSW[1]	oLEDG[1]	10.265
17	iSW[1]	oLEDG[2]	10.125
18	iSW[1]	oLEDG[0]	9.918
19	iSW[0]	oLEDG[1]	9.653
20	iSW[0]	oLEDG[2]	9.515
21	iSW[0]	oLEDG[0]	9.501

Figure 50: 7-3 Compressor Model Maximum Propagation Delay using MUX and XOR Logic Gates [10]

By referring to Figure 49 and Figure 50, it can clearly be seen that the minimum propagation delay and simulated propagation delay has maximum value on input iSW[5] and output oLEDG[0] and maximum value on input iSW[5] and output oLEDG[1] respectively. In this case, the maximum propagation delay on Figure 50 will be considered which the critical path of 7-3 compressor model is the path that allowed signals to pass through from input iSW[5] to output oLEDG[1]. The critical path is the path between the input and output with maximum circuit delay. In order to have better understanding about the 7-3 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 51*.

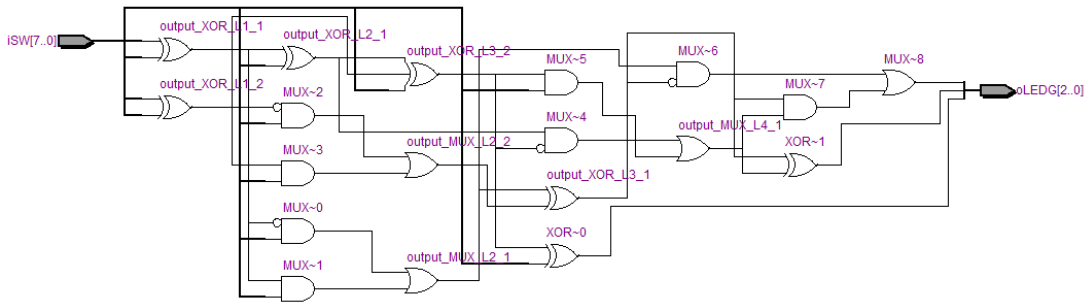


Figure 51: Overall RTL View of MUX-XOR Logic Gates 7-3 Compressor Model [10]

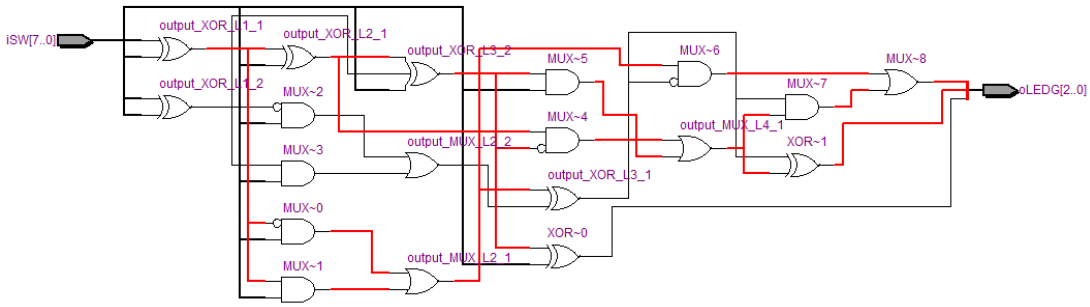


Figure 52: RTL View in Relationship with Input iSW[5] and Output oLEDG[1] [10]

Figure 51 shows the RTL view of 7-3 compressor model using multiplexer and XOR logic gates. As we can see, the compressor model consisted of 8 AND gates, 7 XOR gates, and 4 OR gates. With input iSW[5] and output oLEDG[1] as shown in *Figure 44*, there are 4 AND gates, 5 XOR gates, and 3 OR gate involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, increasing the number of XOR gates will drastically increase the propagation delay of the compressor.

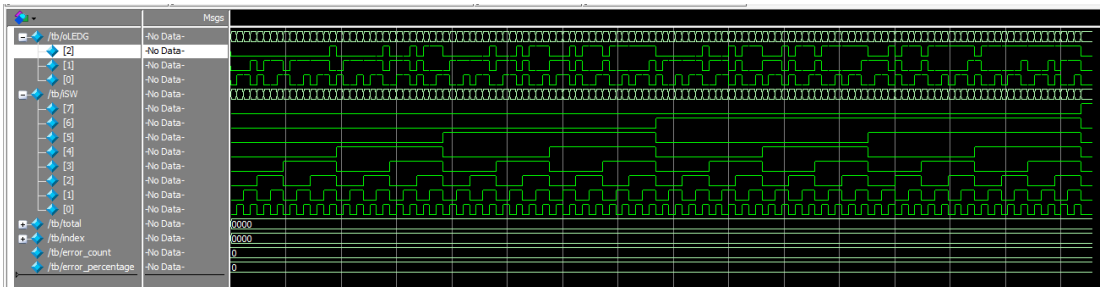


Figure 53: 7-3 Compressor Functional Simulation using ModelSim-Altera

The 7-3 compressor model and self-written functional test bench are written using Verilog Hardware Description Language as shown in Figure 84 and Figure 85 based on Shima Mehrabi, et al. researched proposal [12] which consisted of 4 2-bit Multiplexers and 8 XOR logic gates. The 7-3 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 128 (111111_2) and observed the outputs of 7-3 compressor model ranged from 0 (000_2) to 7 (111_2). *Figure 54* above shows the functional simulation for 7-3 compressor using the self-written test bench with the aids of ModelSim-Altera.

```
# The data input is 126
# Total =  6, oLEDG = 6
# Ok. Result = 6
#
# The data input is 127
# Total =  7, oLEDG = 7
# Ok. Result = 7
#
# Total Error Count = 0.00
# The Failing Rate for Functional Testbench is 0.00
#
```

Figure 54: Error Result of 7-3 Compressor Functional Test bench on Console

Figure 54 above shows the number of error counts and percentage of error of outputs in 7-bits different combinations of inputs for 7-3 compressor model. The above error count is 0 after running 7-bits different combinations of inputs indicating the 7-3 compressor model is designed properly as expected.

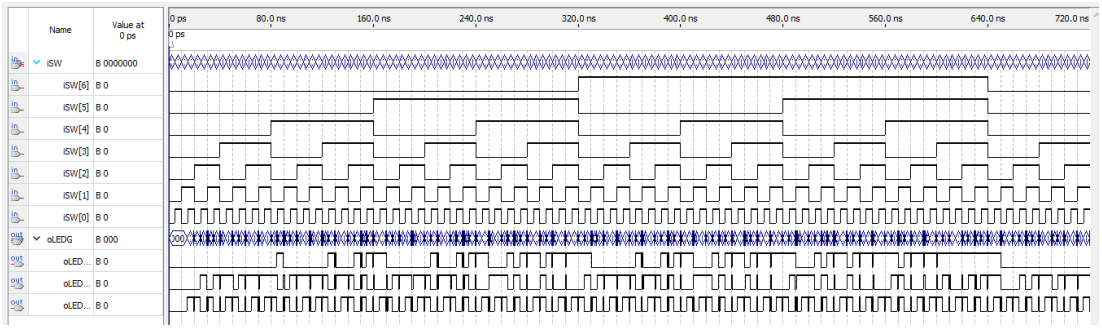


Figure 55: Overall 7-3 MUX-XOR Compressor Model Timing Simulation

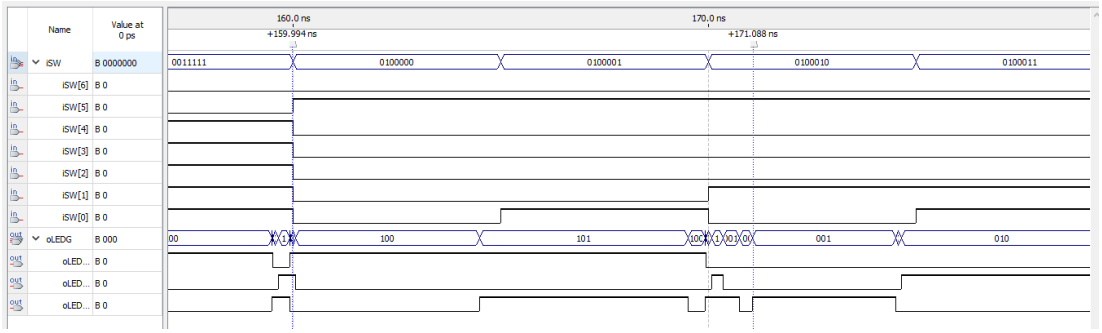


Figure 56: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[1] with bit 1

Figure 55 shows the overall timing simulation for 7-3 compressor model designed. The method of inserting signal for timing simulation is same as previously discussed 7-3 Compressor using Adder Method. *Figure 56* shows the timing simulation for the critical path of 7-3 compressor model. Referring to Figure 50, it can be noticed that input iSW[5] and output oLEDG[1] will have the maximum propagation delay. To prove the statement, *Figure 56* data is obtained and compared in *Table 12*.

Table 12: Propagation Delay of iSW[5] and oLEDG[1] with Input 1s

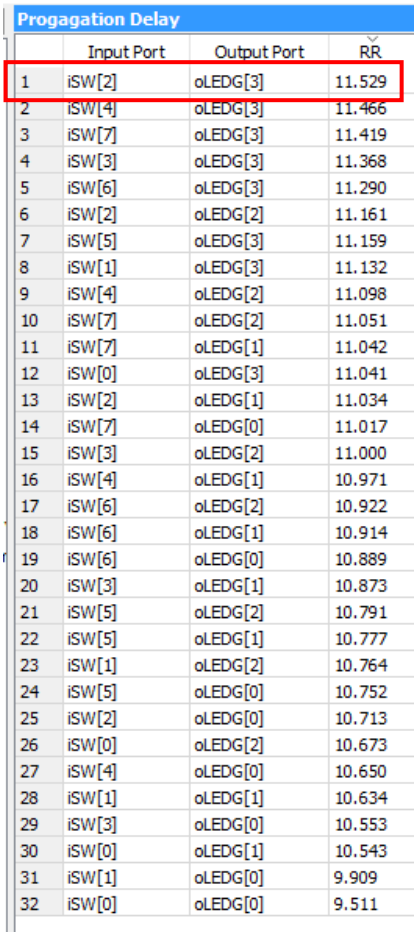
Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Output – Input Transition**	**Percentage Error**
iSW[5]	159.994 ns	171.088 ns	10.985 ns	6.934 %

4.7 8-4 COMPRESSOR (MUX AND HALF ADDERS)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[7]	oLEDG[3]	6.107
2	iSW[6]	oLEDG[3]	6.050
3	iSW[5]	oLEDG[3]	5.991
4	iSW[2]	oLEDG[3]	5.973
5	iSW[7]	oLEDG[0]	5.969
6	iSW[4]	oLEDG[3]	5.965
7	iSW[3]	oLEDG[3]	5.941
8	iSW[6]	oLEDG[0]	5.911
9	iSW[7]	oLEDG[1]	5.896
10	iSW[7]	oLEDG[2]	5.865
11	iSW[5]	oLEDG[0]	5.851
12	iSW[6]	oLEDG[1]	5.839
13	iSW[2]	oLEDG[0]	5.809
14	iSW[6]	oLEDG[2]	5.808
15	iSW[4]	oLEDG[0]	5.801
16	iSW[2]	oLEDG[1]	5.788
17	iSW[4]	oLEDG[1]	5.780
18	iSW[5]	oLEDG[1]	5.780
19	iSW[3]	oLEDG[0]	5.777
20	iSW[2]	oLEDG[2]	5.762
21	iSW[3]	oLEDG[1]	5.756
22	iSW[4]	oLEDG[2]	5.754
23	iSW[5]	oLEDG[2]	5.749
24	iSW[3]	oLEDG[2]	5.730
25	iSW[1]	oLEDG[3]	5.640
26	iSW[1]	oLEDG[0]	5.475
27	iSW[0]	oLEDG[3]	5.470
28	iSW[1]	oLEDG[1]	5.455
29	iSW[1]	oLEDG[2]	5.429
30	iSW[0]	oLEDG[0]	5.306
31	iSW[0]	oLEDG[1]	5.285
32	iSW[0]	oLEDG[2]	5.259

Figure 57: Minimum 8-4 Compressor Model Minimum Propagation Delay using MUX and Half Adders [1]



Progagation Delay

	Input Port	Output Port	RR
1	iSW[2]	oLEDG[3]	11.529
2	iSW[4]	oLEDG[3]	11.466
3	iSW[7]	oLEDG[3]	11.419
4	iSW[3]	oLEDG[3]	11.368
5	iSW[6]	oLEDG[3]	11.290
6	iSW[2]	oLEDG[2]	11.161
7	iSW[5]	oLEDG[3]	11.159
8	iSW[1]	oLEDG[3]	11.132
9	iSW[4]	oLEDG[2]	11.098
10	iSW[7]	oLEDG[2]	11.051
11	iSW[7]	oLEDG[1]	11.042
12	iSW[0]	oLEDG[3]	11.041
13	iSW[2]	oLEDG[1]	11.034
14	iSW[7]	oLEDG[0]	11.017
15	iSW[3]	oLEDG[2]	11.000
16	iSW[4]	oLEDG[1]	10.971
17	iSW[6]	oLEDG[2]	10.922
18	iSW[6]	oLEDG[1]	10.914
19	iSW[6]	oLEDG[0]	10.889
20	iSW[3]	oLEDG[1]	10.873
21	iSW[5]	oLEDG[2]	10.791
22	iSW[5]	oLEDG[1]	10.777
23	iSW[1]	oLEDG[2]	10.764
24	iSW[5]	oLEDG[0]	10.752
25	iSW[2]	oLEDG[0]	10.713
26	iSW[0]	oLEDG[2]	10.673
27	iSW[4]	oLEDG[0]	10.650
28	iSW[1]	oLEDG[1]	10.634
29	iSW[3]	oLEDG[0]	10.553
30	iSW[0]	oLEDG[1]	10.543
31	iSW[1]	oLEDG[0]	9.909
32	iSW[0]	oLEDG[0]	9.511

Figure 58: 8-4 Compressor Model Maximum Propagation Delay using MUX and Half Adders [1]

By referring to Figure 57 and Figure 58, it can clearly be seen that the minimum propagation delay and simulated propagation delay has maximum value on input iSW[7] and output oLEDG[3] and maximum value on input iSW[2] and output oLEDG[3] respectively. In this case, the maximum propagation delay on Figure 58 will be considered which the critical path of 8-4 compressor model is the path that allowed signals to pass through from input iSW[2] to output oLEDG[3]. The critical path is the path between the input and output with maximum circuit delay. In order to have better understanding about the 8-4 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 59*.

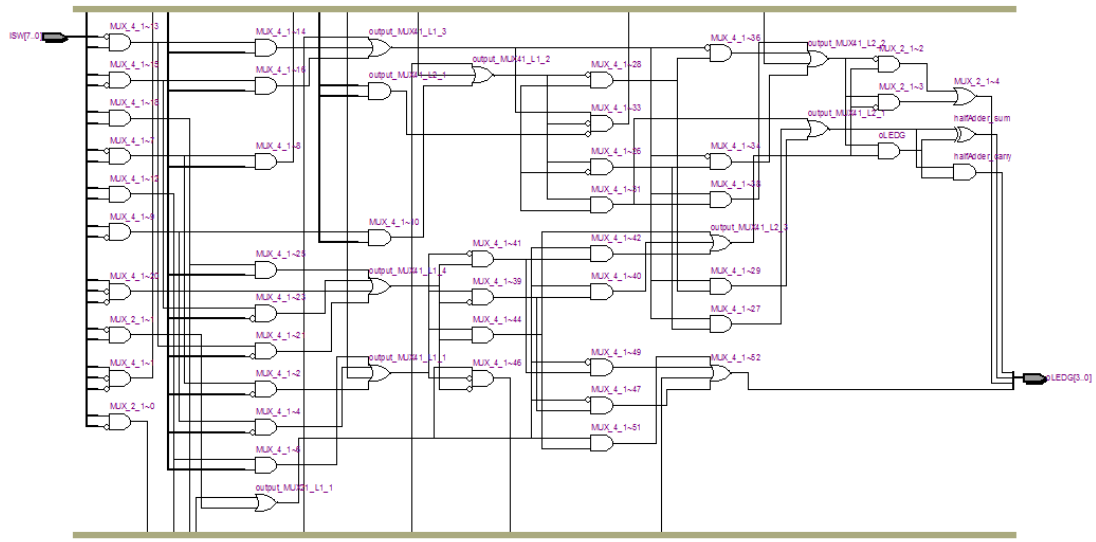


Figure 59: Overall RTL View of MUX and Half Adders' 8-4 Compressor Model [1]

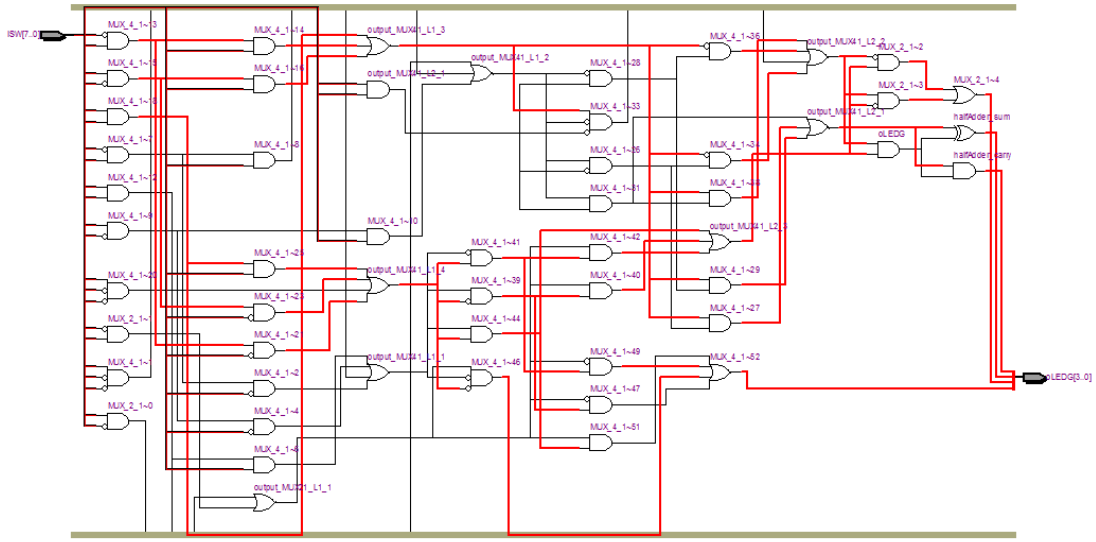


Figure 60: RTL View in Relationship with Input iSW[2] and Output oLEDG[3] [1]

Figure 59 shows the RTL view of 8-4 compressor model using multiplexer and XOR logic gates. As we can see, the compressor model consisted of 43 AND gates, 1 XOR gates, and 10 OR gates. With input iSW[2] and output oLEDG[3] as shown in *Figure 60*, there are 5 AND gates, 1 XOR gates, and 3 OR gate involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay.

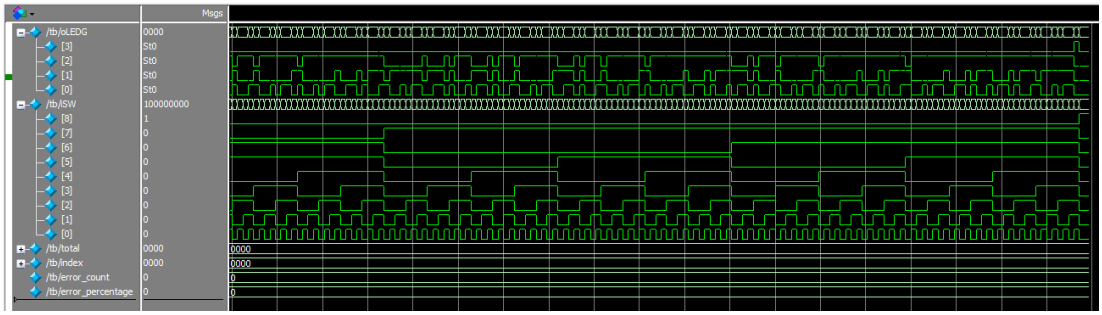


Figure 61: 8-4 Compressor Functional Simulation using ModelSim-Altera

The 8-4 compressor model and self-written functional test bench are written using Verilog Hardware Description Language as shown in Figure 86 and Figure 87 based on Marimuthu, et al. researched proposal [1] which consisted of 8 4-bit Multiplexers, 2 2-bit Multiplexers and 1 half adder. The 8-4 compressor model undergone functional self-written functional test bench which tested with inputs from 0 (0000000_2) to 256 (11111111_2) and observed the outputs of 8-4 compressor model ranged from 0 (000_2) to 8 (1000_2). *Figure 61* above shows the functional simulation for 7-3 compressor using the self-written test bench with the aids of ModelSim-Altera.

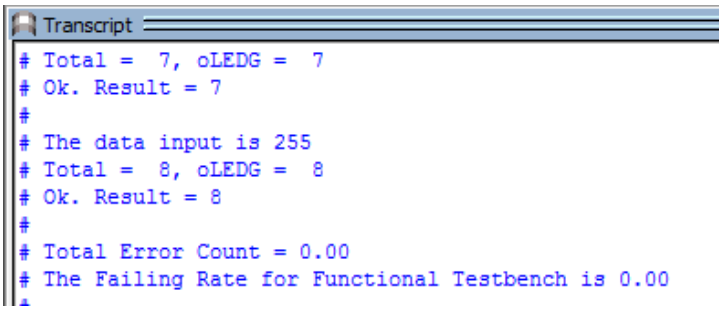


Figure 62: Error Result of 8-4 Compressor Functional Test bench

Figure 62 above shows the number of error counts and percentage of error of outputs in 8-bits different combinations of inputs for 8-4 compressor model. The above error count is 0 after running 8-bits different combinations of inputs indicating the 8-4 compressor model is designed properly as expected.

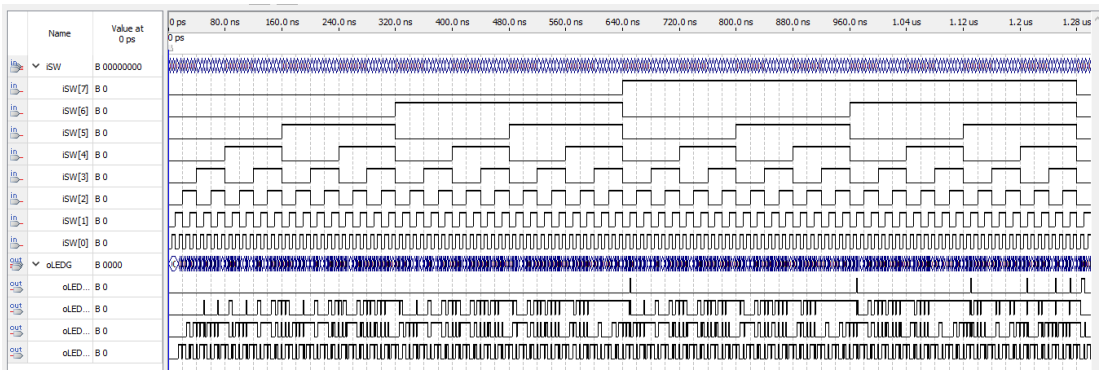


Figure 63: Overall 8-4 MUX-Half Adder Compressor Model Timing Simulation

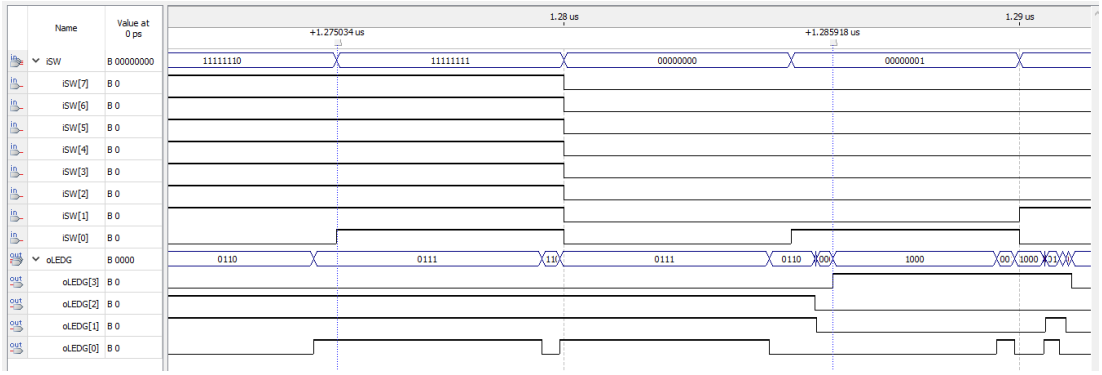


Figure 64: Timing Simulation for Inputs iSW[2] with bit 1 and output oLEDG[3] with bit 1

Figure 63 shows the overall timing simulation for 7-3 compressor model designed. The method of inserting signal for timing simulation is same as previously discussed 8-4 Compressor using Adder Method. *Figure 64* shows the timing simulation for the critical path of 8-4 compressor model. Referring to Figure 58, it can be noticed that input iSW[2] and output oLEDG[3] will have the maximum propagation delay. To prove the statement, *Figure 64* data is obtained and compared in *Table 13*.

Table 13: Propagation Delay of iSW[2] and oLEDG[3] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Time Between Obtained**	**Percentage Error**
iSW[2]	1275.034 ns	1285.918 ns	10.884 ns	0.8536 %

4.8 9-4 COMPRESSOR (MUX AND HALF ADDERS)

Minimum Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[3]	6.106
2	iSW[7]	oLEDG[3]	6.037
3	iSW[6]	oLEDG[3]	5.979
4	iSW[8]	oLEDG[3]	5.924
5	iSW[2]	oLEDG[3]	5.917
6	iSW[4]	oLEDG[3]	5.911
7	iSW[7]	oLEDG[0]	5.892
8	iSW[5]	oLEDG[2]	5.889
9	iSW[3]	oLEDG[3]	5.882
10	iSW[1]	oLEDG[3]	5.873
11	iSW[5]	oLEDG[0]	5.846
12	iSW[7]	oLEDG[2]	5.846
13	iSW[6]	oLEDG[0]	5.835
14	iSW[5]	oLEDG[1]	5.808
15	iSW[6]	oLEDG[2]	5.788
16	iSW[8]	oLEDG[0]	5.777
17	iSW[2]	oLEDG[0]	5.772
18	iSW[8]	oLEDG[2]	5.733
19	iSW[7]	oLEDG[1]	5.732
20	iSW[1]	oLEDG[0]	5.729
21	iSW[2]	oLEDG[2]	5.726
22	iSW[0]	oLEDG[3]	5.707
23	iSW[4]	oLEDG[2]	5.694
24	iSW[1]	oLEDG[2]	5.682
25	iSW[6]	oLEDG[1]	5.675
26	iSW[3]	oLEDG[2]	5.665
27	iSW[4]	oLEDG[0]	5.651
28	iSW[3]	oLEDG[0]	5.622
29	iSW[8]	oLEDG[1]	5.617
30	iSW[4]	oLEDG[1]	5.613
31	iSW[2]	oLEDG[1]	5.611
32	iSW[3]	oLEDG[1]	5.584
33	iSW[1]	oLEDG[1]	5.568
34	iSW[0]	oLEDG[0]	5.562
35	iSW[0]	oLEDG[2]	5.516
36	iSW[0]	oLEDG[1]	5.401

Figure 65: Minimum 9-4 Compressor Model Minimum Propagation Delay using MUX and Half Adder [1]

Progagation Delay

	Input Port	Output Port	RR
1	iSW[5]	oLEDG[3]	11.747
2	iSW[5]	oLEDG[2]	11.428
3	iSW[7]	oLEDG[3]	11.416
4	iSW[4]	oLEDG[3]	11.312
5	iSW[6]	oLEDG[3]	11.290
6	iSW[3]	oLEDG[3]	11.211
7	iSW[2]	oLEDG[3]	11.204
8	iSW[8]	oLEDG[3]	11.144
9	iSW[5]	oLEDG[1]	11.137
10	iSW[1]	oLEDG[3]	11.095
11	iSW[7]	oLEDG[2]	11.088
12	iSW[4]	oLEDG[2]	10.993
13	iSW[6]	oLEDG[2]	10.958
14	iSW[3]	oLEDG[2]	10.892
15	iSW[2]	oLEDG[2]	10.885
16	iSW[7]	oLEDG[0]	10.848
17	iSW[8]	oLEDG[2]	10.819
18	iSW[7]	oLEDG[1]	10.797
19	iSW[1]	oLEDG[2]	10.775
20	iSW[5]	oLEDG[0]	10.751
21	iSW[6]	oLEDG[0]	10.722
22	iSW[0]	oLEDG[3]	10.702
23	iSW[4]	oLEDG[1]	10.702
24	iSW[6]	oLEDG[1]	10.667
25	iSW[2]	oLEDG[0]	10.631
26	iSW[3]	oLEDG[1]	10.601
27	iSW[2]	oLEDG[1]	10.598
28	iSW[8]	oLEDG[0]	10.576
29	iSW[8]	oLEDG[1]	10.528
30	iSW[1]	oLEDG[0]	10.525
31	iSW[1]	oLEDG[1]	10.488
32	iSW[0]	oLEDG[2]	10.383
33	iSW[4]	oLEDG[0]	10.297
34	iSW[3]	oLEDG[0]	10.190
35	iSW[0]	oLEDG[0]	10.129
36	iSW[0]	oLEDG[1]	10.096

Figure 66: 9-4 Compressor Model Maximum Propagation Delay using MUX and Half Adder [1]

By referring to Figure 65 and Figure 66, it can clearly be seen that both the minimum propagation delay and simulated propagation delay has maximum value on input iSW[5] and output oLEDG[3]. In this case, input iSW[5] and output oLEDG[3] will be considered as critical path of 9-4 compressor model. The critical path is the path between the input and output with maximum circuit delay. In order to have better understanding about the 9-4 compressor model, Register Transfer Level (RTL) view of circuit is shown in *Figure 67*.

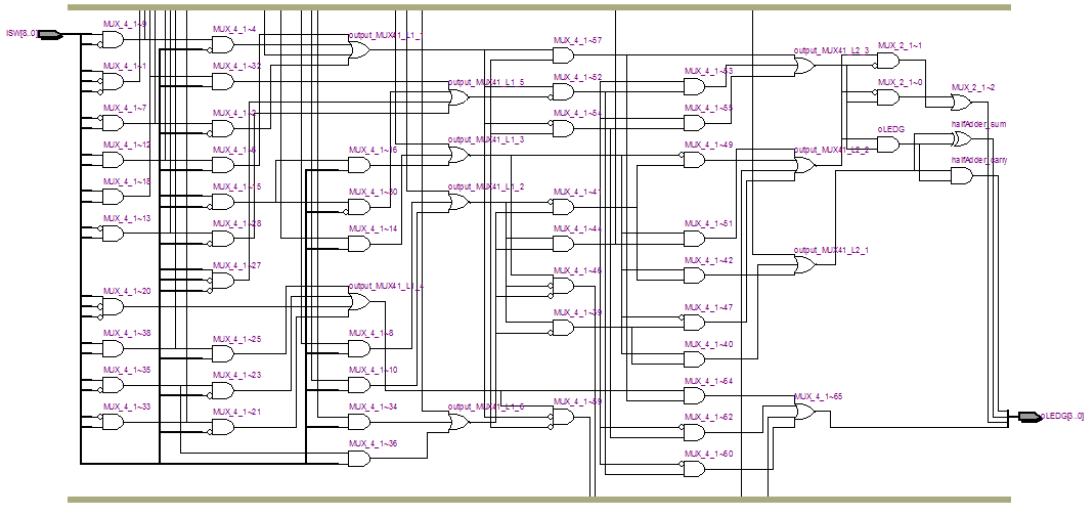


Figure 67: Overall RTL View of MUX-Half Adder 9-4 Compressor Model [1]

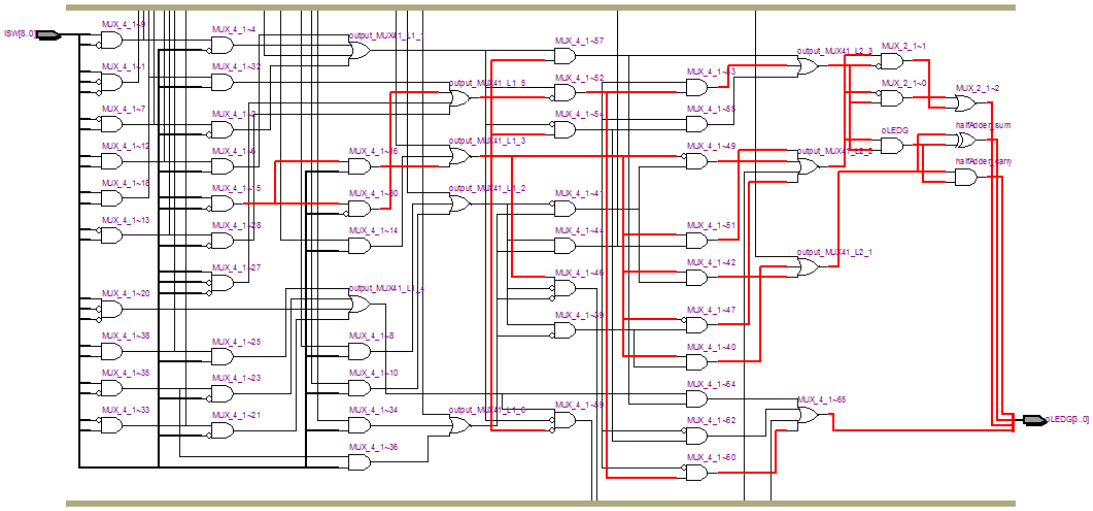


Figure 68: RTL View in Relationship with Input iSW[5] and Output oLEDG[3] [1]

Figure 67 shows the RTL view of 9-4 compressor model using multiplexer and half adder. As we can see, the compressor model consisted of 49 AND gates, 1 XOR gates, and 11 OR gates. With input iSW[5] and output oLEDG[3] as shown in*Figure 68*, there are 4 AND gates, 1 XOR gates, and 2 OR gate involved. In RTL circuit, theoretically, XOR gates will be contributing most of the propagation delay if compared to AND gates and OR gates propagation delay. In other words, when the number of XOR gates increases, the propagation delay of the compressor will be increased dramatically.

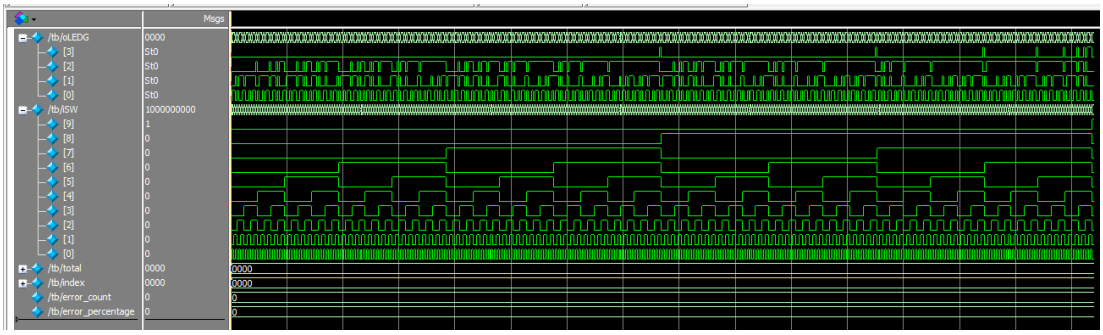


Figure 69: 9-4 Compressor Functional Simulation using ModelSim-Altera

The 7-3 compressor model and self-written functional test bench are written using Verilog HDL as shown in Figure 88 and Figure 89 based on Marimuthu, et al. researched proposal [1] which consisted of 10 4-bit multiplexers, 1 2-bit multiplexers and 1 half adder. *Figure 69* above shows the functional simulation for 9-4 compressor.

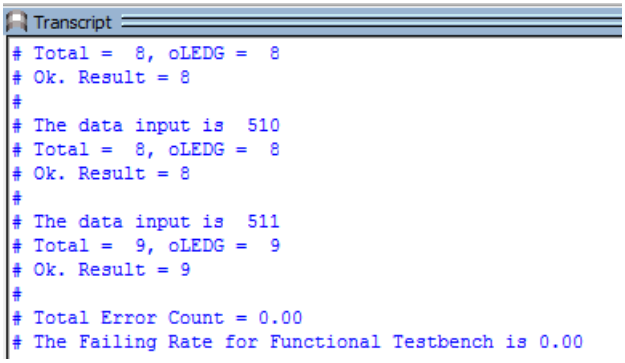


Figure 70: Error Result of 9-4 Compressor Functional Test bench

Figure 70 above shows the number of error counts and percentage of error of outputs in 9-bits different combinations of inputs for 9-4 compressor model. The above error count is 0 after running 9-bits different combinations of inputs indicating the 9-4 compressor model is designed properly as expected.

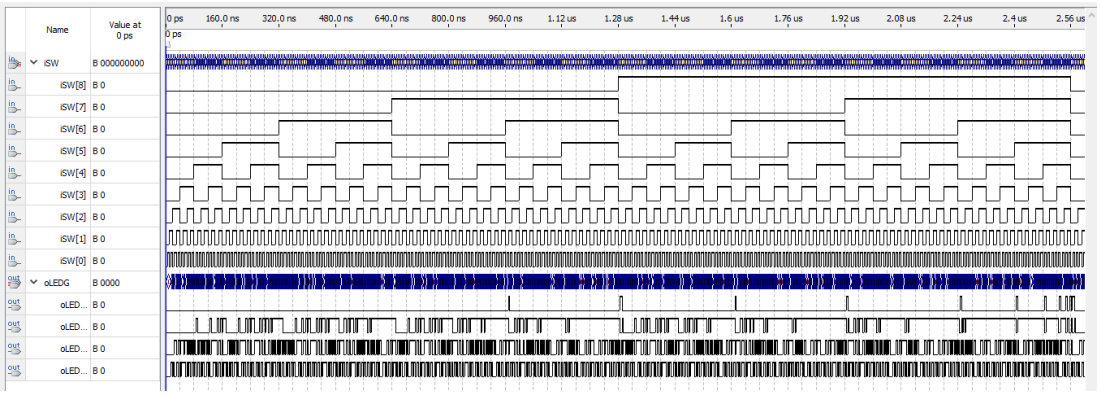


Figure 71: Overall 9-4 MUX-Half Adder Compressor Model Timing Simulation

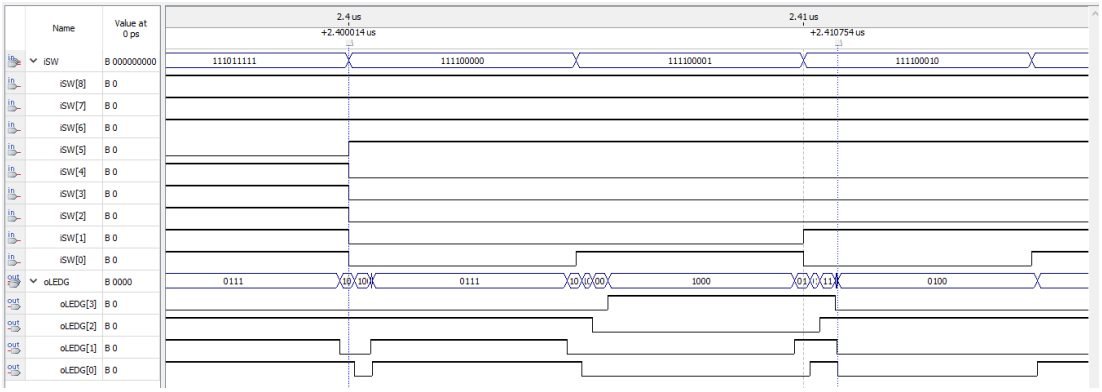


Figure 72: Timing Simulation for Inputs iSW[5] with bit 1 and output oLEDG[3] with bit 1

Figure 71 shows the overall timing simulation for 9-4 compressor model designed. The method of inserting signal for timing simulation is same as previously discussed 8-4 Compressor using Adder Method. *Figure 72* shows the timing simulation for the critical path of 9-4 compressor model. Referring to Figure 66, it can be noticed that input iSW[5] and output oLEDG[3] will have the maximum propagation delay. To prove the statement, *Figure 72* data is obtained and compared in *Table 14*.

Table 14: Propagation Delay of iSW[5] and oLEDG[3] with Input 1s

Input(s) with 1s	**Start Input Transition**	**Start Output Transition**	**Time Between Obtained**	**Percentage Error**
iSW[5]	2400.014 ns	2410.754 ns	10.74 ns	0.4475 %

4.9 DISCUSSIONS

Before going into the collected researched results, it is better to understand the reason to choose performance evaluation of 8-3 compressor in this project. As the order of compressor said, it will have 8-bit inputs and gives 3-bit outputs. Normally, the 3-bit output can only represent decimal digit 0 (000) to 7 (111) but it can't represent decimal digit 8 (1000). However, the most significant bit of 8 can be represented using AND logic gates with fan-in of 8 and this has reduced the output bits' compressor to 3. Besides that, 8-3 compressors design is expected to be more efficient than 8-4 compressor as the output bit of 8-4 compressors which is 4-bit output can be used to represent decimal digit 15 (1111) but it is only used to represent the most significant bit of 4-bit output (1000) hence it under-utilizes the output bits while 8-3 compressors is able to fully utilising all the output bits to represent the input bits while the most significant bit of 8 (1000) can be represented using AND logic gates with fan-in of 8. Figure below shows the simplified concept of using 8-3 compressor.

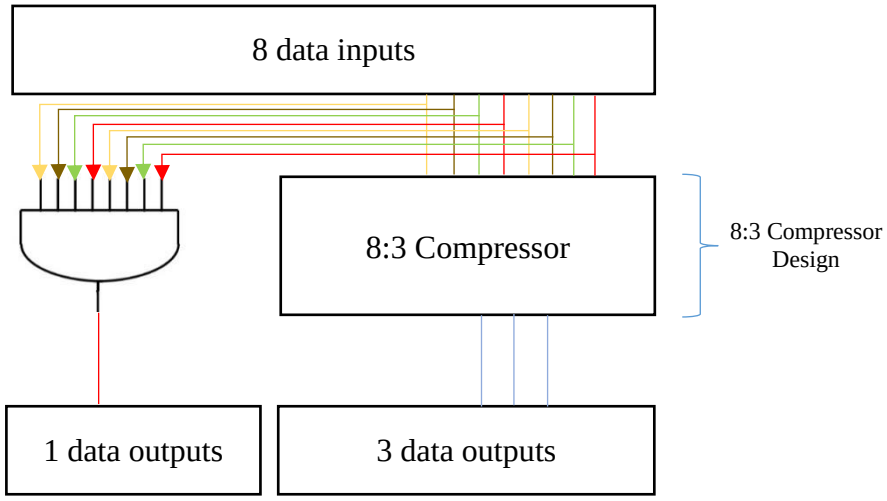


Figure 73: Figure showing Simplified Concept of using 8-3 Compressor

Table 15: Summary of Overall Compressor Types in Relationship with Logic Gate Numbers and Critical Path Propagation Delay

Compressor Order	Compressor Type	Total Number of Logic Gates Used	Critical Path Propagation Delay (ns)
7-3	Adders	19	11.094
	MUX-XOR	20	10.985
8-3	Adders	25	10.453
8-4	Adders	24	11.360
	MUX-XOR	54	10.884
9-3	Adders	28	14.985
9-4	Adders	29	10.010
	MUX-XOR	61	10.740

Table 16: Summary of Overall Compressor Types in Relationship with Cell Area, Power Consumption and Power Delay Product (PDP)

Compressor Order	Compressor Type	Cell Area (μm^2)	Power Consumption (µW)	Power Delay Product (J)
7-3	Adders	89.785	15.885	1.763×10^{-13}
	MUX-XOR	90.238	17.462	2.617×10^{-13}
8-3	Adders	94.782	18.987	1.985×10^{-13}
8-4	Adders	93.139	18.276	2.076×10^{-13}
	MUX-XOR	110.78	19.264	2.097×10^{-13}
9-3	Adders	101.30	20.643	3.093×10^{-13}
9-4	Adders	102.19	21.848	2.187×10^{-13}
	MUX-XOR	112.19	24.459	2.627×10^{-13}

Table 15 and Table 16 show the overall compressor types and its performance. In this table, the different orders of compressor are compared with its types which are MUX-XOR types or adder's types, compressor's total number of logic gates used, compressor's critical path propagation delay, compressor's power delay product, compressor's cell area, and compressor's power consumption. The compressor's total number of logic gates used and compressor's critical path propagation delay data were obtained through creating compressor finite-state machine using Altera Quartus II Web Edition and running functional and timing simulation using ModelSim while the compressor's cell area and compressor's power consumption data were captured via the R. Marimuthu [1], R. Nirlakalla [2], A. Dandapat [3], N. Pokhriyal [7] , and S. Mehrabi [10] research papers. To have better understanding of the comparison table, 7-3, 8-4, and 9-4 compressors will be discussed as below.

Table 17: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 7-3 Compressors

Compressor Type	**Total Number of Logic Gates Used**	**Critical Path Propagation Delay (ns)**
Adders	19	11.094
MUX-XOR	20	10.985

Table 18: Cell Area, Power Consumption and Power Delay Product Comparisons between 7-3 Compressors

Compressor Type	**Cell Area (μm^2)**	**Power Consumption (µW)**	**Power Delay Product (J)**
Adders	89.785	15.885	1.763×10^{-13}
MUX-XOR	90.238	17.462	2.617×10^{-13}

Table *17* shows the total number of logic gates used and critical path propagation delay comparison for 2 different 7-3 researched compressor designs. Table 18 shows the cell area, power consumption and power delay product comparisons between 7-3 Compressors. Through the comparison, it can be seen that the performance of 7-3 Adder compressor is better than 7-3 MUX-XOR compressor as it has 1 fewer number of logic gates used, 0.50% smaller cell area, 9.03% lower power consumption, and 32.63% lower power delay product.

Table 19: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 8-4 Compressors

Compressor Type	**Total Number of Logic Gates Used**	**Critical Path Propagation Delay (ns)**
Adders	24	11.360
MUX-XOR	54	10.884

Table 20: Cell Area, Power Consumption and Power Delay Product Comparisons between 8-4 Compressors

Compressor Type	**Cell Area (μm^2)**	**Power Consumption (µW)**	**Power Delay Product (J)**
Adders	93.139	18.276	2.076×10^{-13}
MUX-XOR	110.78	19.264	2.097×10^{-13}

Table 19 shows the total number of logic gates used and critical path propagation delay comparison for 2 different 8-4 researched compressor designs. Table 20 shows the cell area, power consumption and power delay product comparisons between 8-4 Compressors. Through the comparison, it can be seen that the performance of 8-4 adder's compressor is better than 8-4 MUX-XOR compressor as

it has 30 fewer number of logic gates used, 15.92% smaller cell area, 5.13% lower power consumption, and 1.00% lower power delay product.

Table 21: Total Number of Logic Gates Used and Critical Path Propagation Delay Comparisons between 9-4 Compressors

Compressor Type	**Total Number of Logic Gates Used**	**Critical Path Propagation Delay (ns)**
Adders	29	10.010
MUX-XOR	61	10.740

Table 22: Cell Area, Power Consumption and Power Delay Product Comparisons between 9-4 Compressors

Compressor Type	**Cell Area (μm^2)**	**Power Consumption (µW)**	**Power Delay Product (J)**
Adders	102.19	21.848	2.187×10^{-13}
MUX-XOR	112.19	24.459	2.627×10^{-13}

Table 21 shows the total number of logic gates used and critical path propagation delay comparison for 2 different 9-4 researched compressor designs. Table 22 shows the cell area, power consumption and power delay product comparisons between 9-4 Compressors. Through the comparison, it can be seen that the performance of 9-4 adder's compressor is better than 9-4 MUX-XOR compressor as it has 32 fewer number of logic gates used, 6.80% lower critical path propagation delay, 8.91% smaller cell area, 10.68% lower power consumption, and 16.75% lower power delay product.

After comparing the 7-3, 8-4, and 9-4 compressors design in terms of critical path propagation delay, total number of logic gats used, compressor's cell area, compressor power consumption, and power delay product, a good design has to be referred. For 7-3 compressors, 8-4 compressor and 9-4 compressor designs, it is preferable to use adders instead of combination of multiplexers and XOR gates as it gives minor increases propagation delay while it uses drastically less logic gates than combinations of multiplexers and adders, lower compressor's power consumption, smaller compressor cell area, and lower power delay product. Therefore, for 8-3 compressor design, it is preferable to use adder's compressor instead of a combination of multiplexers and XOR logic gates as it shall give lower power delay product, also known as switching energy, saving costs, resources, and power consumption.

From the above comparison, it can be noticed that the higher the order of compressor, the more logic gates needed to be used, which will lead to higher power consumption and higher critical propagation delay. Since it has only 8 inputs, it is sufficient to use 8-3 compressor with 8-input fan-in AND logic gate instead of 8-4 compressor so that the output of 8:3 compressor can be fully utilized, saving logic gates used and reducing power delay product.

CHAPTER 5: CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION

In a nutshell, the main purpose of having this project is to repeat a few researched designs to verify their performance report using Altera Quartus II Web Edition and ModelSim-Altera software, implementing it using FPGA and compared among the researched designs to verify which is the best method to design 8-3 compressor. After studying and comparing the 7-3, 8-4, and 9-4 compressor researched designs by considering the combination of multiplexers and XOR logic gates, combination of multiplexers and adders, and adders by comparing their logic gates numbers and propagation delay, it can be concluded that using adder's method is the better way to design the 8-3 compressors as it uses fewer numbers of logic gates, having lower cell area, and lower propagation delay. Using fewer number of logic gates will lead to lowering the total power consumption needed and smaller cell area of compressor. The reason of using FPGA in this project is FPGA can be easily programmed to perform complex combinational logic functions. Besides that, it supports timing simulation which can emulate an implementation into real devices to verify the implemented design meets all functional and timing requirements to ensure the finished design is free of defects.

5.2 RECOMMENDATIONS

A few recommendations proposed to this project are shown as below:

- Design 8:3 compressors and implement it on field programmable gate array (FPGA) to evaluate the design's fan-in, energy delay product (EDP), power delay product (PDP), and cell area in order to have better handling the multiplication operations

REFERENCES

[1] R. Marimuthu, D. Bansal, S. Balamurugan, and P. Mallick, "DESIGN OF 8-4 AND 9-4 COMPRESSORS FORHIGH SPEED MULTIPLICATION," *American Journal of Applied Sciences,* vol. 10, p. 893, 2013.

[2] R. Nirlakalla, R. T. Subba, and T. Jayachandra-Prasad, "Performance evaluation of high speed compressors for high speed multipliers," *Serbian Journal of Electrical Engineering,* vol. 8, pp. 293-306, 2011.

[3] A. Dandapat, S. Ghosal, P. Sarkar, and D. Mukhopadhyay, "A 1.2-ns16× 16-Bit Binary Multiplier Using High Speed Compressors," *International Journal of Electrical and Electronics Engineering,* vol. 4, p. 3, 2010.

[4] Nidhi Pokhriyal, Harsimranjit Kaur, Dr. Neelam Rup Prakash, "Compressor Based Area-Efficient Low-Power 8x8 Vedic Multiplier," *Nidhi Pokhriyal et al Int. Journal of Engineering Research and Applications,* vol. 3, no. 6, pp. 1469-1472, Nov-Dec 2013.

[5] S. Veeramachaneni, K. M. Krishna, L. Avinash, S. R. Puppala, and M. Srinivas, "Novel architectures for high-speed and low-power 3-2, 4-2 and 5-2 compressors," *20th International Conference on VLSI Design held jointly with 6th International Conference on Embedded Systems (VLSID'07),* pp. 324-329, 2007.

[6] A. Dandapat, S. Ghosal, P. Sarkar, and D. Mukhopadhyay, "HIGH SPEED LOW POWER CARRY SAVE 6-3 COMPRESSOR FOR HIGH SPEED MULTIPLIER APPLICATION".

[7] N. Pokhriyal, H. Kaur, and D. N. Prakash, "Compressor Based Area-Efficient Low-Power 8x8 Vedic Multiplier," *Int. Journal of Engineering Research and Applications,* vol. 3, pp. 1469-1472, 2013.

[8] M. Mehta, V. Parmar, and E. Swartzlander, "High-speed multiplier design using multi-input counter and compressor circuits," *10th IEEE Symposium,* pp. 43-50, 1991.

[9] Mayur Mehta, Vijay Parmar, Earl Swartzlander, "High-Speed Multiplier Design Using Multi-Input Counter and Compressor Circuits," in *Computer Arithmetic, 1991. Proceedings., 10th IEEE Symposium on*, 1991.

[10] S. Mehrabi, K. Navi, and O. Hashemipour, "Performance analysis and simulation of two different architectures of (6: 3) and (7: 3) compressors based on carbon Nano-Tube Field Effect Transistors," in *Nanoelectronics Conference (INEC), 2013 IEEE 5th International*, 2013.

[11] Ienne, A. K. Verma and P., "Automatic synthesis of compressor trees: reevaluating large counters," *Proceedings of the conference on Design, automation and test in Europe,* pp. 443-448, 2007.

[12] S. Mehrabi, R. F. Mirzaee, S. Zamanzadeh, and A. Jamalian, "A New Hybrid 16-Bit x 16-Bit Multiplier Architecture by m: 2 and m: 3 Compressors," *International Journal of Information and Electronics Engineering,* vol. 6, p. 79, 2016.

[13] Jonathon Rose, Abbas EL Gamal, and Alberto Sangiovanni-Vincentelli, "Architecture of Field-Programmable," *Architecture of Field-Programmable,* vol. 81, no. 7, July, 1993.

[14] M. Rouholamini, O. Kavehie, A.-P. Mirbaha, S. J. Jasbi, and K. Navi, "A new design for 7: 2 compressors," in *2007 IEEE/ACS International Conference on Computer Systems and Applications*, 2007.

[15] Shima Mehrabi, Reza Faghih Mirzaee, Sharareh Zamanzadeh, Keivan Navi, and Omid Hashemipour, "Design, analysis, and implementation of partial product reduction phase by using wide m:3 ($4 \leq m \leq 10$) compressors," *Int. J. High Performance Systems Architecture,* vol. 4, pp. 231-241, 2013.

APPENDIX

```
/*======================== 8:3 Compressors Algorithms ================================================*/
//Inputs for middle full adder
assign input_middle_sum_1 = halfAdder_sum(iSW[0], iSW[1]);
assign input_middle_sum_2 = fullAdder_sum(iSW[2], iSW[3], iSW[4]);
assign input_middle_sum_3 = fullAdder_sum(iSW[5], iSW[6], iSW[7]);

assign input_middle_carry_1 = halfAdder_carry(iSW[0], iSW[1]);
assign input_middle_carry_2 = fullAdder_carry(iSW[2], iSW[3], iSW[4]);
assign input_middle_carry_3 = fullAdder_carry(iSW[5], iSW[6], iSW[7]);

//Inputs for middle half adder
assign input_half_sum_1 = fullAdder_sum(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_half_sum_2 = 0;

assign input_full_1 = halfAdder_carry(input_half_sum_1, input_half_sum_2);
assign input_full_2 = fullAdder_carry(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_full_3 = fullAdder_sum(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);
assign input_full_4 = fullAdder_carry(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);
assign input_full_5 = fullAdder_carry(input_full_1, input_full_2, input_full_3);

//Outputs
assign oLEDG[0] = halfAdder_sum(input_half_sum_1, input_half_sum_2);
assign oLEDG[1] = fullAdder_sum(input_full_1, input_full_2, input_full_3);
assign oLEDG[2] = halfAdder_sum(input_full_4, input_full_5);
```

Figure 74: 8-3 Compressor Model / Finite State Machine

```
initial begin
    // Functional Testbench
    for (iSW = 9'h0; iSW < 9'h100; iSW = iSW + 9'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 256;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h8; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            3'b000  :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end
                            else if (total == 4'h8) begin $display("Ok. Result = 8\n"); end else show_error();
            3'b001  :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            3'b010  :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            3'b011  :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            3'b100  :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            3'b101  :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            3'b110  :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            3'b111  :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
endtask
```

Figure 75: 8-3 Compressor Functional Test bench

```
/*======================== 8:4 Compressors Algorithms ================================================*/
//Inputs for Middle Full Adders
assign input_middle_sum_1 = halfAdder_sum(iSW[0], iSW[1]);
assign input_middle_sum_2 = fullAdder_sum(iSW[2], iSW[3], iSW[4]);
assign input_middle_sum_3 = fullAdder_sum(iSW[5], iSW[6], iSW[7]);

assign input_middle_carry_1 = halfAdder_carry(iSW[0], iSW[1]);
assign input_middle_carry_2 = fullAdder_carry(iSW[2], iSW[3], iSW[4]);
assign input_middle_carry_3 = fullAdder_carry(iSW[5], iSW[6], iSW[7]);

// Inputs for Final Full Adders
assign input_full_sum_1   = fullAdder_sum(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_full_carry_1 = fullAdder_carry(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_full_sum_2   = fullAdder_sum(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);
assign input_full_carry_2 = fullAdder_carry(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);

//Outputs
assign oLEDG[0]             = input_full_sum_1;
assign oLEDG[1]             = halfAdder_sum(input_full_carry_1, input_full_sum_2);
assign input_final_half = halfAdder_carry(input_full_carry_1, input_full_sum_2);
assign oLEDG[2]             = halfAdder_sum(input_final_half, input_full_carry_2);
assign oLEDG[3]             = halfAdder_carry(input_final_half, input_full_carry_2);
```

Figure 76: 8-4 Compressor Model / Finite State Machine

```
initial begin
    // Functional Testbench
    for (iSW = 9'h0; iSW < 9'h100; iSW = iSW + 9'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 256;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h8; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            4'h0    :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            4'h1    :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            4'h2    :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            4'h3    :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            4'h4    :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            4'h5    :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            4'h6    :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            4'h7    :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            4'h8    :  if (total == 4'h8) begin $display("Ok. Result = 8\n"); end else show_error();
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
endtask
```

Figure 77: 8-4 Compressor Functional Test bench

```
/*======================= 9:4 Compressors Algorithms ================================================*/
//Inputs for Middle Full Adders
assign input_middle_sum_1 = fullAdder_sum(iSW[0], iSW[1], iSW[2]);
assign input_middle_sum_2 = fullAdder_sum(iSW[3], iSW[4], iSW[5]);
assign input_middle_sum_3 = fullAdder_sum(iSW[6], iSW[7], iSW[8]);

assign input_middle_carry_1 = fullAdder_carry(iSW[0], iSW[1], iSW[2]);
assign input_middle_carry_2 = fullAdder_carry(iSW[3], iSW[4], iSW[5]);
assign input_middle_carry_3 = fullAdder_carry(iSW[6], iSW[7], iSW[8]);

// Inputs for Final Full Adders
assign input_full_sum_1   = fullAdder_sum(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_full_carry_1 = fullAdder_carry(input_middle_sum_1, input_middle_sum_2, input_middle_sum_3);
assign input_full_sum_2   = fullAdder_sum(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);
assign input_full_carry_2 = fullAdder_carry(input_middle_carry_1, input_middle_carry_2, input_middle_carry_3);

/*==================== Full and Half Adders Functions ============================================*/
//Outputs
assign oLEDG[0]            = input_full_sum_1;
assign oLEDG[1]            = halfAdder_sum(input_full_carry_1, input_full_sum_2);
assign input_final_half = halfAdder_carry(input_full_carry_1, input_full_sum_2);
assign oLEDG[2]            = halfAdder_sum(input_final_half, input_full_carry_2);
assign oLEDG[3]            = halfAdder_carry(input_final_half, input_full_carry_2);
```

Figure 78: 9-4 Compressor Model / Finite State Machine

```
initial begin
    // Functional Testbench
    for (iSW = 10'h0; iSW < 10'h200; iSW = iSW + 10'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 512;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h9; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            4'h0    :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            4'h1    :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            4'h2    :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            4'h3    :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            4'h4    :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            4'h5    :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            4'h6    :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            4'h7    :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            4'h8    :  if (total == 4'h8) begin $display("Ok. Result = 8\n"); end else show_error();
            4'h9    :  if (total == 4'h9) begin $display("Ok. Result = 9\n"); end else show_error();
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
```

Figure 79: 9-4 Compressor Functional Test bench

```
/*======================== 9:3 Compressors Algorithms ================================================*/
//Inputs for level 1 full adder
assign input_level1_sum_1 = fullAdder_sum(iSW[0], iSW[1], iSW[2]);
assign input_level1_sum_2 = fullAdder_sum(iSW[3], iSW[4], iSW[5]);
assign input_level1_sum_3 = fullAdder_sum(iSW[6], iSW[7], iSW[8]);

assign input_level1_carry_1 = fullAdder_carry(iSW[0], iSW[1], iSW[2]);
assign input_level1_carry_2 = fullAdder_carry(iSW[3], iSW[4], iSW[5]);
assign input_level1_carry_3 = fullAdder_carry(iSW[6], iSW[7], iSW[8]);

//Inputs for level 2 full adder
assign input_level2_sum_1 = fullAdder_sum(input_level1_sum_1, input_level1_sum_2, input_level1_sum_3);
assign input_level2_sum_2 = fullAdder_sum(input_level1_carry_1, input_level1_carry_2, input_level1_carry_3);

assign input_level2_carry_1 = fullAdder_carry(input_level1_sum_1, input_level1_sum_2, input_level1_sum_3);
assign input_level2_carry_2 = fullAdder_carry(input_level1_carry_1, input_level1_carry_2, input_level1_carry_3);

assign input_full_carry = halfAdder_carry(0, input_level2_sum_1);

//Outputs
assign oLEDG[0] = halfAdder_sum(0, input_level2_sum_1);
assign oLEDG[1] = fullAdder_sum(input_full_carry, input_level2_carry_1, input_level2_sum_2);
assign oLEDG[2] = fullAdder_carry(input_full_carry, input_level2_carry_1, input_level2_sum_2) ^ input_level2_carry_2;
```

Figure 80: 9-3 Compressor Model / Finite State Machine

```
initial begin
   // Functional Testbench
   for (iSW = 10'h0; iSW < 10'h200; iSW = iSW + 10'h1) begin
      #3 $display("The data input is %d",iSW);
      #3 run();
   end
   $display("Total Error Count = %.2f",error_count);
   #10 error_percentage = error_count * 100 / 512;
   $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
   error_count = 0;
   #1 $finish;
end

task run();
   begin
      for (index = 4'h0; index < 4'h9; index = index + 4'h1) begin
         if (iSW[index] == 1) begin
            total = total + 4'h1;
         end
      end
      $display("Total = %d, oLEDG = %d", total,oLEDG);
      case (oLEDG)
         3'h0  :  if (total == 4'h0 || total == 4'h8) begin $display("Ok. Result = 0\n"); end else show_error();
         3'h1  :  if (total == 4'h1 || total == 4'h9) begin $display("Ok. Result = 1\n"); end else show_error();
         3'h2  :  if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
         3'h3  :  if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
         3'h4  :  if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
         3'h5  :  if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
         3'h6  :  if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
         3'h7  :  if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
         default  :  show_error();
      endcase
      //$display("Total = %d",total);
      total = 0;
      index = 0;
   end
endtask
```

Figure 81: 9-3 Compressor Functional Test bench

```
/*======================== 7:3 Compressors Algorithms ================================================*/
//Inputs for middle full adder
assign input_middle_sum_1 = fullAdder_sum(iSW[1], iSW[2], iSW[3]);
assign input_middle_sum_2 = fullAdder_sum(iSW[4], iSW[5], iSW[6]);

assign input_middle_carry_1 = fullAdder_carry(iSW[1], iSW[2], iSW[3]);
assign input_middle_carry_2 = fullAdder_carry(iSW[4], iSW[5], iSW[6]);

//Inputs for middle half adder
assign input_half_carry_1 = fullAdder_carry(iSW[0], input_middle_sum_1, input_middle_sum_2);

//Outputs
assign oLEDG[0] = fullAdder_sum(iSW[0], input_middle_sum_1, input_middle_sum_2);
assign oLEDG[1] = fullAdder_sum(input_half_carry_1, input_middle_carry_1, input_middle_carry_2);
assign oLEDG[2] = fullAdder_carry(input_half_carry_1, input_middle_carry_1, input_middle_carry_2);
```

Figure 82: 7-3 Compressor Model / Finite-State Machine

```
initial begin
    // Functional Testbench
    for (iSW = 8'h0; iSW < 8'h80; iSW = iSW + 8'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 128;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h8; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            3'h0    :   if (total == 3'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            3'h1    :   if (total == 3'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            3'h2    :   if (total == 3'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            3'h3    :   if (total == 3'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            3'h4    :   if (total == 3'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            3'h5    :   if (total == 3'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            3'h6    :   if (total == 3'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            3'h7    :   if (total == 3'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
endtask
```

Figure 83: 7-3 Compressor Functional Test bench

```
/*======================== 7:3 Compressors Algorithms ================================================*/
//Level 1
assign output_XOR_L1_1 = XOR(iSW[6], iSW[5]);
assign output_XOR_L1_2 = XOR(iSW[3], iSW[2]);

//Level 2
assign output_MUX_L2_1 = MUX(iSW[6], iSW[4], output_XOR_L1_1);
assign output_XOR_L2_1 = XOR(output_XOR_L1_1, iSW[4]);
assign output_MUX_L2_2 = MUX(iSW[3], iSW[1], output_XOR_L1_2);
assign output_XOR_L2_2 = XOR(output_XOR_L1_2, iSW[1]);

//Level 3
assign output_XOR_L3_1 = XOR(output_MUX_L2_1, output_MUX_L2_2);
assign output_XOR_L3_2 = XOR(output_XOR_L2_1, output_XOR_L2_2);

//Level 4
assign output_MUX_L4_1 = MUX(output_XOR_L2_1, iSW[0], output_XOR_L3_2);
assign output_XOR_L4_1 = XOR(output_XOR_L3_2, iSW[0]);

//Outputs
assign oLEDG[0] = output_XOR_L4_1;
assign oLEDG[1] = XOR(output_XOR_L3_1, output_MUX_L4_1);
assign oLEDG[2] = MUX(output_MUX_L2_1, output_MUX_L4_1, output_XOR_L3_1);
```

Figure 84: 7-3 Compressor Model / Finite-State Machine using MUX-XOR Logic Gates

```
initial begin
    // Functional Testbench
    for (iSW = 8'h0; iSW < 8'h80; iSW = iSW + 8'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 128;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h8; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            3'h0    :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            3'h1    :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            3'h2    :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            3'h3    :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            3'h4    :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            3'h5    :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            3'h6    :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            3'h7    :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
endtask
```

Figure 85: MUX-XOR Logic Gates 7-3 Compressor Functional Test bench

```
/*======================== 8:4 Compressors Algorithms ================================================*/
//Level 1

assign output_MUX41_L1_1 = MUX_4_1(iSW[5], ~iSW[5], ~iSW[5], iSW[5], iSW[6], iSW[7]);
assign output_MUX41_L1_2 = MUX_4_1(0, iSW[5], iSW[5], 1, iSW[6], iSW[7]);
assign output_MUX41_L1_3 = MUX_4_1(0, iSW[2], iSW[2], 1, iSW[3], iSW[4]);
assign output_MUX21_L1_1 = MUX_2_1(iSW[0], ~iSW[0], iSW[1]);
assign output_MUX41_L1_4 = MUX_4_1(iSW[2], ~iSW[2], ~iSW[2], iSW[2], iSW[3], iSW[4]);

//Level 2
assign output_MUX41_L2_1 = MUX_4_1(0, output_MUX41_L1_3, output_MUX41_L1_3, 1, (iSW[0] & iSW[1]), output_MUX41_L1_2);
assign output_MUX41_L2_2 = MUX_4_1(output_MUX41_L1_3, ~output_MUX41_L1_3, ~output_MUX41_L1_3, output_MUX41_L1_3, (iSW[0] & iSW[1]), output_MUX41_L1_2);
assign output_MUX41_L2_3 = MUX_4_1(0, output_MUX21_L1_1, output_MUX21_L1_1, 1, output_MUX41_L1_4, output_MUX41_L1_1);

//Outputs
assign oLEDG[0] = MUX_4_1(output_MUX21_L1_1, ~output_MUX21_L1_1, ~output_MUX21_L1_1, output_MUX21_L1_1, output_MUX41_L1_4, output_MUX41_L1_1);
assign oLEDG[1] = MUX_2_1(output_MUX41_L2_3, ~output_MUX41_L2_3, output_MUX41_L2_2);
assign oLEDG[2] = halfAdder_sum((output_MUX41_L2_2 & output_MUX41_L2_3), output_MUX41_L2_1);
assign oLEDG[3] = halfAdder_carry((output_MUX41_L2_2 & output_MUX41_L2_3), output_MUX41_L2_1);
```

Figure 86: 8-4 Compressor Model / Finite-State Machine using MUX and Half Adders

```
initial begin
    // Functional Testbench
    for (iSW = 9'h0; iSW < 9'h100; iSW = iSW + 9'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 256;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h9; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            4'h0    :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            4'h1    :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            4'h2    :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            4'h3    :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            4'h4    :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            4'h5    :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            4'h6    :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            4'h7    :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            4'h8    :  if (total == 4'h8) begin $display("Ok. Result = 8\n"); end else show_error();
            default :   show_error();
        endcase
        //$display("Total = %d",total);
        total = 0;
        index = 0;
    end
endtask
```

Figure 87: MUX and Half Adders 8-4 Compressor Functional Test bench

```
/*======================== 9:4 Compressors Algorithms ================================================*/
//Level 1

assign output_MUX41_L1_1 = MUX_4_1(iSW[6], ~iSW[6], ~iSW[6], iSW[6], iSW[7], iSW[8]);
assign output_MUX41_L1_2 = MUX_4_1(0, iSW[6], iSW[6], 1, iSW[7], iSW[8]);
assign output_MUX41_L1_3 = MUX_4_1(0, iSW[3], iSW[3], 1, iSW[4], iSW[5]);
assign output_MUX41_L1_4 = MUX_4_1(iSW[0], ~iSW[0], ~iSW[0], iSW[0], iSW[1], iSW[2]);
assign output_MUX41_L1_5 = MUX_4_1(iSW[3], ~iSW[3], ~iSW[3], iSW[3], iSW[4], iSW[5]);
assign output_MUX41_L1_6 = MUX_4_1(0, iSW[0], iSW[0], 1, iSW[1], iSW[2]);

//Level 2
assign output_MUX41_L2_1 = MUX_4_1(0, output_MUX41_L1_3, output_MUX41_L1_3, 1, output_MUX41_L1_6, output_MUX41_L1_2);
assign output_MUX41_L2_2 = MUX_4_1(output_MUX41_L1_3, ~output_MUX41_L1_3, ~output_MUX41_L1_3, output_MUX41_L1_3, output_MUX41_L1_3, output_MUX41_L1_6, output_MUX41_L1_2);
assign output_MUX41_L2_3 = MUX_4_1(0, output_MUX41_L1_4, output_MUX41_L1_4, 1, output_MUX41_L1_5, output_MUX41_L1_1);

//Outputs
assign oLEDG[0] = MUX_4_1(output_MUX41_L1_4, ~output_MUX41_L1_4, ~output_MUX41_L1_4, output_MUX41_L1_4, output_MUX41_L1_5, output_MUX41_L1_1);
assign oLEDG[1] = MUX_2_1(output_MUX41_L2_3, ~output_MUX41_L2_3, output_MUX41_L2_2);
assign oLEDG[2] = halfAdder_sum((output_MUX41_L2_2 & output_MUX41_L2_3), output_MUX41_L2_1);
assign oLEDG[3] = halfAdder_carry((output_MUX41_L2_2 & output_MUX41_L2_3), output_MUX41_L2_1);
```

Figure 88: 9-4 Compressor Model / Finite-State Machine using MUX and Half Adders

```
initial begin
    // Functional Testbench
    for (iSW = 10'h0; iSW < 10'h200; iSW = iSW + 10'h1) begin
        #3 $display("The data input is %d",iSW);
        #3 run();
    end
    $display("Total Error Count = %.2f",error_count);
    #10 error_percentage = error_count * 100 / 512;
    $display("The Failing Rate for Functional Testbench is %.2f\n\n", error_percentage);
    error_count = 0;
    #1 $finish;
end

task run();
    begin
        for (index = 4'h0; index < 4'h9; index = index + 4'h1) begin
            if (iSW[index] == 1) begin
                total = total + 4'h1;
            end
        end
        $display("Total = %d, oLEDG = %d", total,oLEDG);
        case (oLEDG)
            4'h0    :   if (total == 4'h0) begin $display("Ok. Result = 0\n"); end else show_error();
            4'h1    :   if (total == 4'h1) begin $display("Ok. Result = 1\n"); end else show_error();
            4'h2    :   if (total == 4'h2) begin $display("Ok. Result = 2\n"); end else show_error();
            4'h3    :   if (total == 4'h3) begin $display("Ok. Result = 3\n"); end else show_error();
            4'h4    :   if (total == 4'h4) begin $display("Ok. Result = 4\n"); end else show_error();
            4'h5    :   if (total == 4'h5) begin $display("Ok. Result = 5\n"); end else show_error();
            4'h6    :   if (total == 4'h6) begin $display("Ok. Result = 6\n"); end else show_error();
            4'h7    :   if (total == 4'h7) begin $display("Ok. Result = 7\n"); end else show_error();
            4'h8    :  if (total == 4'h8) begin $display("Ok. Result = 8\n"); end else show_error();
            4'h9    :  if (total == 4'h9) begin $display("Ok. Result = 9\n"); end else show_error();
            default :   show_error();
        endcase
        total = 0;
        index = 0;
    end
endtask
```

Figure 89: MUX and Half Adders 9-4 Compressor Functional Test bench