

FPGA IMPLEMENTATION OF ARITHMETIC OPERATIONS USING QUATERNARY SIGNED DIGIT

Dissertation submitted in partial fulfillment of the requirement for the degree of

MASTERS OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING

By

RADHIKA

(152016)

UNDER THE GUIDANCE OF

Dr. SHRUTI JAIN



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

May 2017

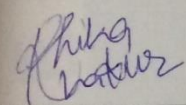
TABLE OF CONTENTS

	PAGE NUMBER
DECLARATION BY THE SCHOLAR	II
SUPERVISOR'S CERTIFICATE	III
PREFACE AND ACKNOWLEDGEMENT	IV
ABSTRACT	V
LIST OF ACRONYMS AND ABBREVIATIONS	VI
LIST OF SYMBOLS	VII
LIST OF FIGURES	VIII
LIST OF TABLES	IX
CHAPTER 1	
INTRODUCTION	1
1.1 QUATERNARY SIGNED DIGIT NUMBER	2
2.1 THESIS ORGANIZATION	5
CHAPTER 2	
LITERATURE REVIEW	6
CHAPTER 3	
ADDER AND SUBTRACTOR	
3.1 BCD ADDER	10
3.2.1 BCD ADDER DESIGN	11
3.2.2 SIMULATION	13
3.2.3 RESULT	15

3.2	QSD ADDER	15
	3.3.1 QSD ADDER DESIGN	15
	3.3.2 DESIGN ALGORITHM	17
	3.3.3 SIMULATION	23
	3.3.4 RESULT	25
3.3	QSD SUBTRACTOR	26
	3.4.1 DESIGN ALGORITHM	26
	3.4.2 SIMULATION	29
	3.4.3 RESULTS	31
3.4	COMPARISON	32
CHAPTER 4		
MULTIPLIER		33
4.1	WALLACE TREE MULTIPLIER	33
	4.2.1 WALLACE TREE DESIGN	33
	4.2.2 SIMULATION	34
	4.2.3 RESULT	36
4.2	BAUGH WOOLEY MULTIPLIER	36
	4.3.1 SIMULATION	38
	4.3.2 RESULT	40
4.3	QUATERNARY SIGNED DIGIT MULTIPLIER	40
	4.4.1 DESIGN ALGORITHM	40
	4.4.2 SIMULATION	45
	4.4.3 RESULT	47
4.4	COMPARISON	48
CONCLUSION AND FUTURE WORK		50
REFERENCES		51

DECLARATION BY THE SCHOLAR

We hereby declare that the work reported in the M-Tech thesis entitled "**FPGA IMPLEMENTATION OF ARITHMETIC OPERATIONS USING QUATERNARY SIGNED DIGIT**" submitted at **Jaypee University of Information Technology, Wagnaghat India**, is an authentic record of my work carried out under the supervision of **Dr. SHRUTI JAIN**. We have not submitted this work elsewhere for any other degree or diploma.



RADHIKA
(152016)

Department of Electronics & Communication Engineering
Jaypee University of Information Technology, Wagnaghat , India
Date: 29-04-2017



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established by H.P. State Legislative vide Act No. 14 of 2002)
P.O. Wagnaghat, Teh. Kandaghat, Distt. Solan - 173234 (H.P.) INDIA

Website: www.juit.ac.in
Phone No. (91) 01792-257999
Fax: +91-01792-245362

CERTIFICATE

This is to certify that the work reported in the M.tech project report entitled "**FPGA IMPLEMENTATION OF ARITHMETIC OPERATIONS USING QUATERNARY SIGNED DIGIT**" which is being submitted by **Radhika** in fulfillment for the award of Bachelor of Technology in Electronics and Communication Engineering by the Jaypee University of Information Technology, is the record of candidate's own work carried out by him/her under my supervision. This work is original and has not been submitted partially or fully anywhere else for any other degree or diploma.

Shruti Jain
01/05/17

Dr. Shruti Jain

Associate Professor

Department of Electronics & Communication Engineering

Jaypee University of Information Technology, Wagnaghat,

विद्या तत्व ज्योतिसमः

ACKNOWLEDGEMENT

I wish to express my profound gratitude and indebtedness to **Dr. SHRUTI JAIN** , Associate Professor, Department of Electronics and Communication , Jaypee University of Information Technology, Waknaghat, Solan, for introducing the present topic and for their inspiring guidance and valuable suggestion for this thesis work.

Lastly , I feel immense pleasure to express my sincere gratitude to my parents and friends who have always been source of encouragement and strength.

ABSTRACT

The primary problem in digital world is to minimize the area and increase the speed of operations. Using efficient techniques we can overcome these problems. In digital circuits arithmetic operations are very useful and important. Digital circuits are used in microcomputers, signal processing and many other digital systems. Digital system use binary number system which has two states. Binary is represented by binary digit '1' and '0', which represents the two different voltage levels HIGH and LOW. The HIGH voltage level is used to represent 1 and LOW voltage is used to represent 0. A microprocessor is a VLSI device that performed arithmetic operations and many other operations. The microprocessor is used as the central processing unit in microcomputers system so speed of the microprocessor depends upon the maximum speed to accomplish the operations. However, propagation time delay, and circuit complexities are the crucial problems in arithmetic operations. For quick results in digital processor we have to increase the speed of the operation. In binary number system, generation of carry in arithmetic operations create delay problems, reduce the speed of microcomputers and increase the complexity of circuits. The problems of arithmetic operations can be overcome by using Quaternary Signed Digit (QSD) number system rather than binary number system. QSD is a higher radix number system, it implies that higher radix number system is less complex than the lower radix number system. QSD is represented by four decimal number 0, 1, 2 and 3. QSD perform carry free addition, borrow free subtraction and multiplication which reduce the complexity of circuit and has less delay than other arithmetic circuits.

LIST OF ACRONYMS & ABBREVIATIONS

BCD	Binary Coded Decimal
CSA	Carry Save Adder
FPGA	Field Programmable Gate Array
IC	Intermediate Carry
IS	Intermediate Sum
LUT	Look Up Table
QSD	Quaternary Signed Digit
VLSI	Very Large Scale Integration

LIST OF SYMBOLS

b_i	Base of any number
D	Decimal number
X_i	Value from QSD set number

LIST OF FIGURES

Figure Number	Caption	Page Number
1.1	Positive number representation	3
1.2	Negative number representation	3
1.3	Voltage level of QSD	4
3.1	Block diagram of BCD adder	12
3.2	Output of BCD adder	13
3.3	Schematic diagram of BCD adder	14
3.4	n digit QSD adder	16
3.5	Basic concept of QSD adder	17
3.6	Output wave form of QSD	24
3.7	Schematic diagram of QSD adder	24
3.8	Derive successfully programmed on FPGA board	25
3.9	Output wave form of QSD subtraction	30

3.10	Schematic diagram of QSD subtraction	31
4.1	Wallace tree multiplier diagram	34
4.2	Output wave form of Wallace multiplier	34
4.3	Schematic diagram of Wallace multiplier	35
4.4	Baugh Wooley multiplier	37
4.5	Output wave form of Baugh Wooley	38
4.6	Schematic diagram of Baugh Wooley	39
4.7	Flow chart of QSD multiplication	41
4.8	Output wave form of QSD multiplication	45
4.9	Schematic view of QSD multiplier	46
4.10	Derive successfully programmed on FPGA board	47

LIST OF TABLES

Table Number	Caption	Page Number
1.1	Binary and decimal representation	2
3.1	Valid BCD representation	10
3.2	Valid and invalid representation of BCD between(10-15)	11
3.3	Design parameters of BCD	14
3.4	Possible maximum and minimum values of addition	17
3.5	All possible addition combination of two QSD number	19
3.6	Possible values of IC and IS	22
3.7	Design parameter of QSD addition	25
3.8	All possible combination of two QSD subtraction	27
3.9	Design parameter of QSD subtraction	31
3.10	Comparing the results of BCD and QSD adder	32
4.1	Design parameter of Wallace multiplier	36

4.2	Design parameter of Baugh Wooley	40
4.3	Show all possible values of multiplication of two QSD number	41
4.4	Possible output combination between -9 to +9	43
4.5	Design parameter of QSD multiplier	47
4.6	Comparison of QSD multiplier with different multiplier	48
4.7	Comparison of proposed design with the existing design	49

CHAPTER 1

INTRODUCTION

Arithmetic operations has important role in digital electronic. Digital systems are widely used over the analog because it has several advantages. One of advantage is the fast arithmetic calculation which increases the speed of operation in digital system. Digital systems are highly used in data processing, signal system, control system, computation and measurements. There are several techniques to accomplish the arithmetic operation in digital system. One of them is Binary Signed Digit Number, it allow the carry limited operation but with complex addition process. So arithmetic operations have to compromise with the speed of operation, limiting number of bits, delay and circuit complexity. These are the limitation of arithmetic operation in digital system. Digital system performance depends upon the computational speed of arithmetic operations. The growing worldliness of application continuously pushes the design and manufacturing of electronic system to new level of complexity. So the major challenge for the VLSI designers is to reduce the area of the chip without affecting the performance of the system. Because small size leads to advantages in speed and power consumption, since it has small resistance capacitance and inductance. Designer has to use the logical techniques to reduce the area of chip and increase the performance of system. In today's life millions of instruction done by the microprocessors in seconds. Microprocessor is a digital device in microcomputers that can be programmed to perform arithmetic and logic operation and other functions. So designer has to increase the speed of operations so that the microprocessors complete million of instructions fast. As reducing the number of component, the power supply requirement and so on will reduce the system cost also. We generally overcome the arithmetic operation problem of high complexity, delay and less speed of operations. For arithmetic operation using the binary number system which generate carry and increase the usage of storage in circuit. To overcome this problem we are using higher radix number system. Quaternary Signed Digit (QSD) number is a higher radix number system than binary number system. So we implemented different adders and multipliers using XILINX 14.7 ISE and programmed successfully on SPARTAN 3E 100or250 FPGA board.

1.1 QUATERNARY SIGNED DIGIT NUMBER

Quaternary Signed digit number system is a higher radix number system. To perform the arithmetic operations we use QSD Number system. QSD offers the carry free addition, borrow free subtraction and multiplication. It helps microprocessors to accomplish the task with less complexity and time delay. Microcomputers speed depends upon the operational speed. QSD is the number system with radix 4 and it is represented by 4 numbers 0, 1, 2, and 3. Signed digit is represented by $\bar{3}$, $\bar{2}$ and $\bar{1}$ where $\bar{3}$ is -3, $\bar{2}$ is -2 and $\bar{1}$. Whole QSD number set is representing as $\{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$. Normally most of the arithmetic operation deals in decimal number so decimal number D can be represented in form of QSD number system as

$$D = \sum_{i=0}^{n-1} X_i b^i \quad (1)$$

Where X_i is any value from the set of $\{\bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3\}$ and b is base of any number system. In Quaternary Signed Digit number system b is 4 . Now the equation is rewrite with the base number as

$$D = \sum_{i=0}^{n-1} X_i 4^i \quad (2)$$

In QSD +3 is the maximum digit and -3 is the minimum digit as shown in TABLE 1.1.

Table 1.1: Binary and Decimal Representation of QSD

DECIMAL NO.	QSD NO.	BINARY REPRESENTATION
-3	-3	101
-2	-2	110
-1	-1	111
0	0	000

1	1	001
2	2	010
3	3	011

It required 3 bit to represent single decimal number, in which most significant bit represent the sign of the number whether the number is positive or negative while other 2 bits representing decimal number. If signed bit is '0' then number is positive shown in figure 1.1 and if sign bit is '1' then number is negative shown in figure 1.2.

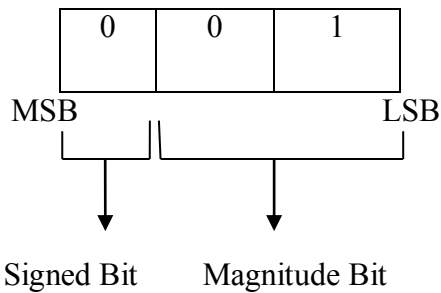


Figure 1.1: Positive number representation.

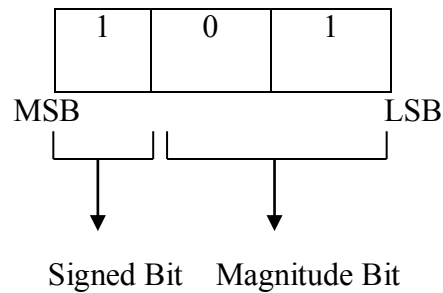


Figure 1.2: Negative number representation.

Quaternary is overcome the problem of binary arithmetic. Binary operation is limited due to carry generation that mean it need more interconnects in VLSI which occupy large area and increase the complexity of the circuit. In VLSI 70% of area is dedicated to interconnects, 20% dedicated to insulation and the remaining 10% is for device. For arithmetic operation if the radix of numeral system is large than it required less number of digit to present the quantity. As discussed QSD logic representation is 0, 1, 2 and 3 so it uses four voltage levels as shown in figure 1.1.

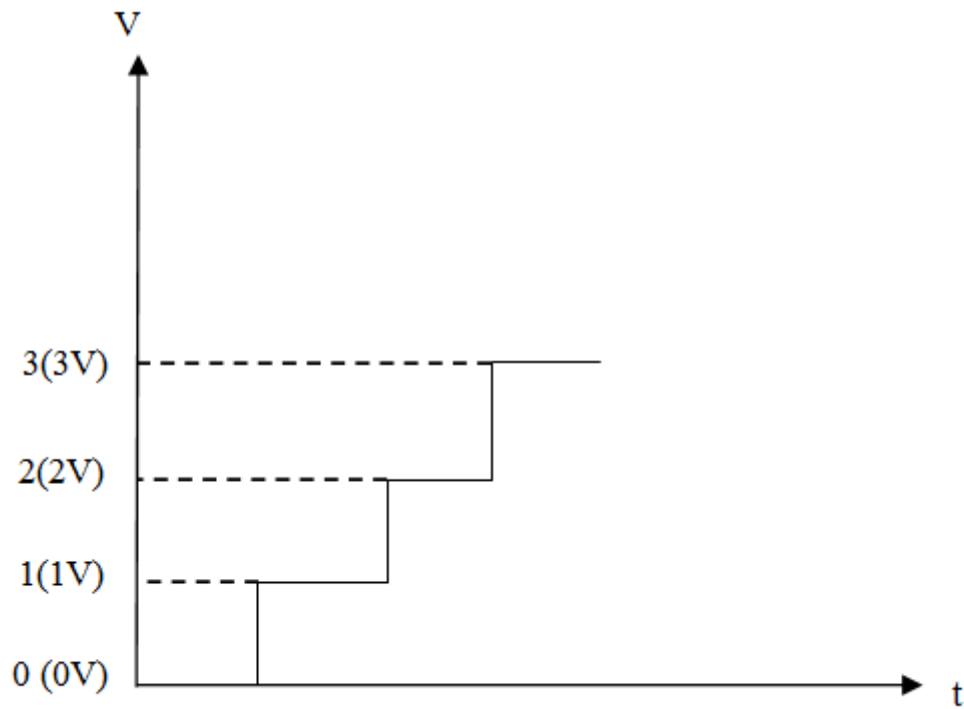


Figure 1.3: Voltage levels of QSD

1.2 THESIS ORGANIZATION

This report is tidy in four chapters.

Chapter 1 of thesis gives the brief introduction of problem in arithmetic operations. This chapter also provides a brief introduction of QSD number system.

In *chapter 2* we introduce literature review, we summarized the objectives of existing publication on arithmetic operations.

In *chapter 3* we implement different types of adders such as BCD and QSD adder. We also implement QSD subtractor in this chapter. We discuss design algorithm of adders and subtractor along with the design parameters and simulation results. We also comparing the results of QSD adder with the BCD.

In *chapter 4* we implement different types of multiplier such as QSD multiplier, Wallace multiplier and Baugh Wooley multiplier. We compare the simulation results and design parameters of these multipliers.

Final is the conclusion of the thesis work and provides recommendation for future work.

CHAPTER 2

LITERATURE REVIEW

We have studied different papers.

1. Krishna M.N. and Ravisekhar T.,(2008)[5],conducted study for fast arithmetic operation with QSD using Verilog HDL. This paper proposed a circuit of QSD which consume low power and energy.
2. Rani R., Singh L. K., Sharma N., (2009)[3], conducted study for FPGA implementation of fast adders using Quaternary signed digit number system. This paper proposed a design of QSD addition and multiplication which will increase the operation speed. Design QSD number uses 25% less space than BSD to store number, higher number of gates can be tolerated for further improvement of QSD adder. BSD tolerance is low to store higher number of gates.
3. Chattopadhyay T. and Sarkar T.,(2012)[11], conducted study for the logical design of quaternary signed digit conversion circuit and its effectuation using operational amplifier In this paper, they reported a new and easy method for conversion from binary number (2's compliment representation) to quaternary-signed digit (QSD).
4. Sahastrabudhey S.B., Bogawar K. M. (2012)[19], conducted study for arithmetical operations in quaternary system using VHDL .The QSD adder is better than other binary adders in terms of number of gates and higher number of bits addition within constant time. Efficient design for adder block to perform addition or multiplication will increase operation speed.
5. Shende P.Y. and Dr. Kshirsagar R.V(2013)[17], conducted study for the quaternary adder design using VHDL. The proposed QSD adder is better than other binary adders in terms of

number of gates and higher number of bits addition within constant time. Efficient design for adder block to perform addition or multiplication will increase operation speed.

6. Dubey S., Rani R., Kumari S., Sharma N. (2013)[3] , conducted study for VLSI implementation of fast addition using quaternary signed digit number system. Implementation for single digit addition, the dynamic power dissipation is 36.255W at 5GHz frequency. These circuits consume less energy and less energy and power, and shows better performance. The delay of the proposed design is 2ns.
7. Suneetha M. and Kumar S.N., Sivakrishna P.(2014)[16],conducted study for design and implementation of 2-digit adder using Quaternary signed digit number system. Implementation for 2 digit addition, the dynamic power dissipation 0.079mW at 5GHz frequency. These circuits consume less energy and less energy and power, and shows better performance. The delay of the proposed design is less
8. Mohan V. and Mohan K. M.,(2014)[6], conducted study for the implementation of quaternary signed adder system. Proposed QSD adder is better than other binary adders in terms of number of gates and higher number of bits addition within constant time QSD number uses less space than BSD.
9. Paisa S. and Babu K.S.,(2014)[13], conducted study for the designing of QSD number system for arithmetic operations. Implementation for single digit addition, the dynamic power dissipation is 36.255—Wat 5GHz frequency. These circuits consume less energy and less energy and power, and shows better performance. The delay of the proposed design is 2ns.
10. Reddy T.R. and Reddy C.V.V.,(2014)[21], conducted study for the high speed arithmetic operations using quaternary signed digit number system. With the use of Quaternary SDN system one can perform different arithmetic operations like addition, subtraction,

multiplication for single digit numbers, with this his number system one can perform high speed operations with better performance. The delay is less.

11. Manasa G., Rao M.D., Miranji K.,(2014)[2], conducted study for the design and analysis of fast addition mechanism for integers using quaternary signed digit number system .The technique for conversion from binary number system to quaternary number system is explained with examples. For the addition of the Quaternary number a different technique is used i.e., for the addition of two QSD numbers first the intermediate sum and carry are generated and then these intermediate sum and carry are added with each other with the help of full adders. The result of this addition is always carrying free.
12. Ramya E. and Deepti P.R.,(2015)[12], conducted study for the implementation of fast addition using quaternary signed digit number system. The implementation of QSD addition, subtraction and multiplication are verified and compared with the Ripple carry adder. The test confirms the superior performance of the QSD adder implementation over other adders beyond 64-bits due to the carry-free addition scheme.
13. Babu M.K, Karthik C., Venkatesh C.,(2015)[14], conducted study for implementation of carry free arithmetic operations by QSDN. The proposed QSD adder is better than other binary adders in terms of number of gates and higher number of bits addition within constant time. Efficient design for adder block to perform addition or multiplication will increase operation speed. QSD number uses 25% less space than BSD.
14. Reddy M.R. and Reddy M.R., (2015) [20] conducted study for Implementation of fast addition with QSD using Verilog. Implementation for single digit addition, the dynamic power dissipation is 36.255W at 5GHz frequency. These circuits consume less energy and less energy and power, and shows better performance. The delay of the proposed design is 2ns.

15. Reddy K.C.S and DR. Rao D.V.,(2015)[1] conducted study for high performance quaternary arithmetic logic unit on programmable logic device, The Proposed QSD arithmetic logic unit is better than other binary arithmetic unit in terms of number of gates and higher number of bits operation with in constant time. Efficient design for adder block to perform addition and multiplication will increase operation speed.QSD number uses 25% less space than BCD to store number, higher number of gates can be tolerated for further improvement of QSD adder.

CHAPTER 3

ADDERS AND SUBTRACTOR

Arithmetic circuits are important part of VLSI technology. Switching techniques of arithmetic system is very high due this it consume more power than others. Arithmetic includes addition, subtraction and multiplication. Now in this chapter we implementing different types of adders and comparing results.

3.1 BCD ADDDER

BCD is Binary Coded Decimal number and also called Excess 3 number system. BCD needed 4 bit to represent decimal number 0-9. In 4 bit representation 1st 10 combinations is valid in BSD as shown in TABLE 3.1.

TABLE 3.1: Valid BCD representation

DECIMAL NO.	BCD REPRESENTATION
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

After 10 combination of BCD 6 combinations are invalid and their valid and invalid representation shows in TABLE 3.2.

TABLE 3.2: Valid and invalid representation of BCD between (10-15).

DECIMAL NO	INVALID BCD	VALID BCD
10	1010	00010000
11	1011	00010001
12	1100	00010010
13	1101	00010011
14	1110	00010100
15	1111	00010101

To represent a single digit number BCD required 4 bits if to represent two digit number then it required 8 bits. In two digit number four bits represent first digit and next four bit represents second digit.

3.2.1 BCD ADDER DESIGN

As shown in block diagram in figure 3.1, 4bits of addend and 4 bits of augends are added in 4 bit of binary block. After addition the results will be of 4 bits S0, S1, S2 and S3. If the sum of the BCD adder is up to 9 then result is representing by 4 bits.

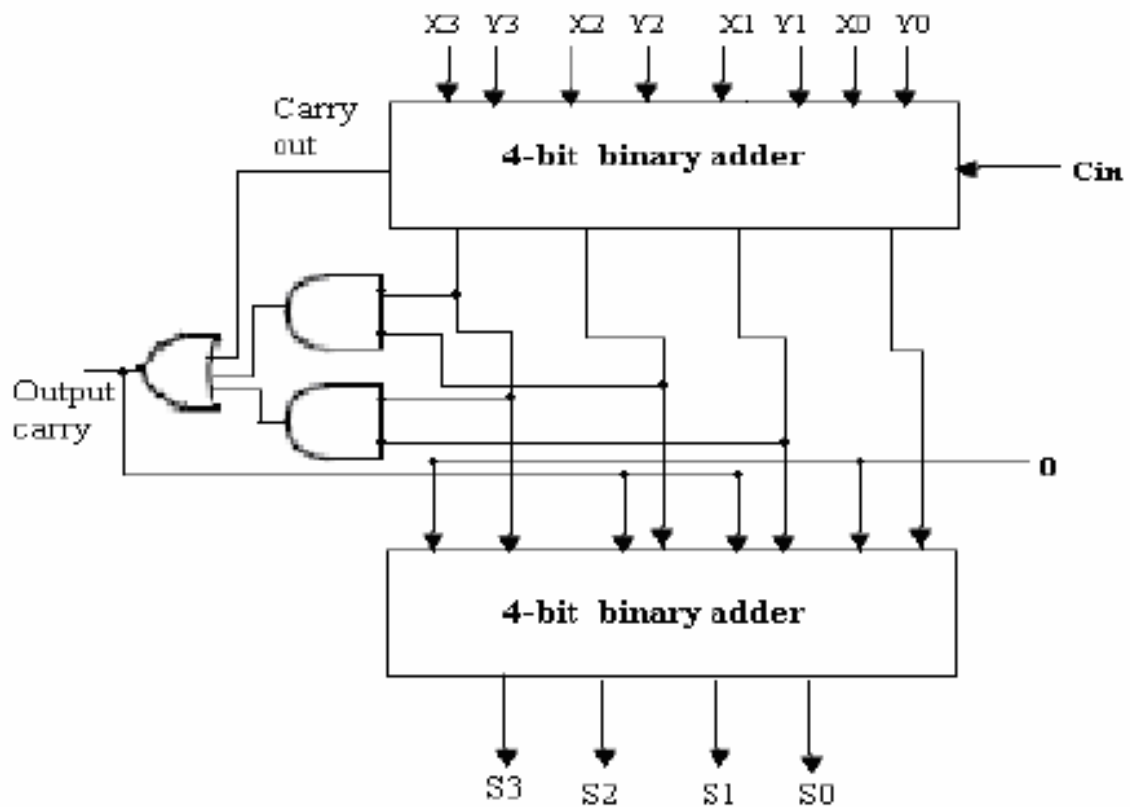


Figure 3.1: Block diagram of BCD adder [27].

If result sum is greater than 9 then we add 6 to the sum so that the sum is valid BCD and it represent by 8 bits as shown in table 3. Let's take an example of BCD adder.

Example 3.1 Add two numbers 6 and 5 in BCD adder.

Solution:

$$\begin{array}{r}
 (6)_{10} = 0110 \\
 (5)_{10} = 0101 \\
 \hline
 \text{Sum} = 1011 \quad \leftarrow \text{invalid BCD number}
 \end{array}$$

Adding 6 to the invalid sum

$$\begin{array}{r}
 1011 \\
 0110 \\
 \hline
 \end{array}$$

Sum=0001 0001 ← valid BCD

In example we add 6 and 5, after adding the BCD result is invalid because the sum of addition is greater than 9. So in next step we add 6 to sum so that it will be a BCD number. In final sum carry represented by 4 bits and rest of sum represent by another 4 bits. Total sum of BCD adder represented by 8 bits

3.2.2 SIMULATION

The code for BCD adder is written in Verilog and the design is simulated and analyzed using Xilinx 14.7 ISE Simulator. Figure 3.2 shows the output waveform of BCD and figure 3.3 shows the schematic diagram. In figure 3.2 we are adding $A[3:0] = 1001$ and $B[3:0] = 1001$ after addition we get sum of 0010 with carry 1. Next step is to add 6 to the previous sum (0010+0110) we get in result is 1000 ,now adding the carry to this result the final sum of BCD adder is 1001.

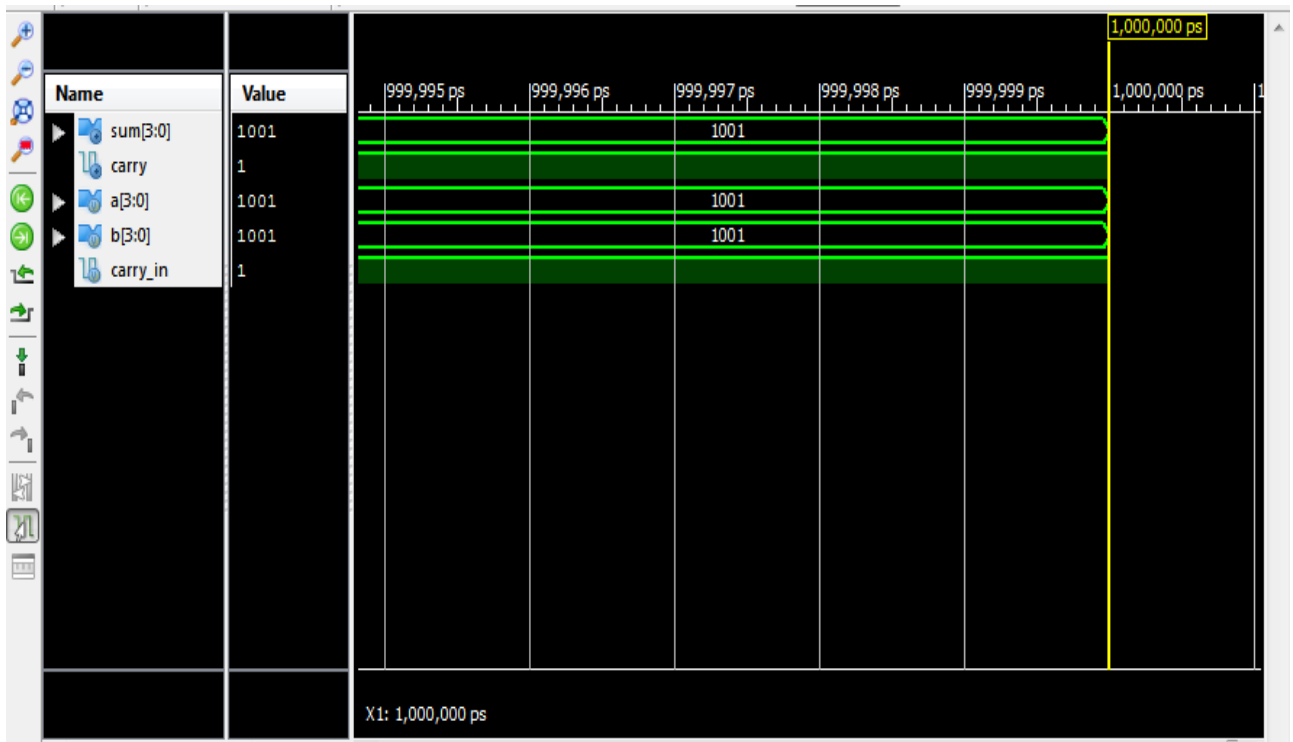


Figure 3.2: Output of BCD adder

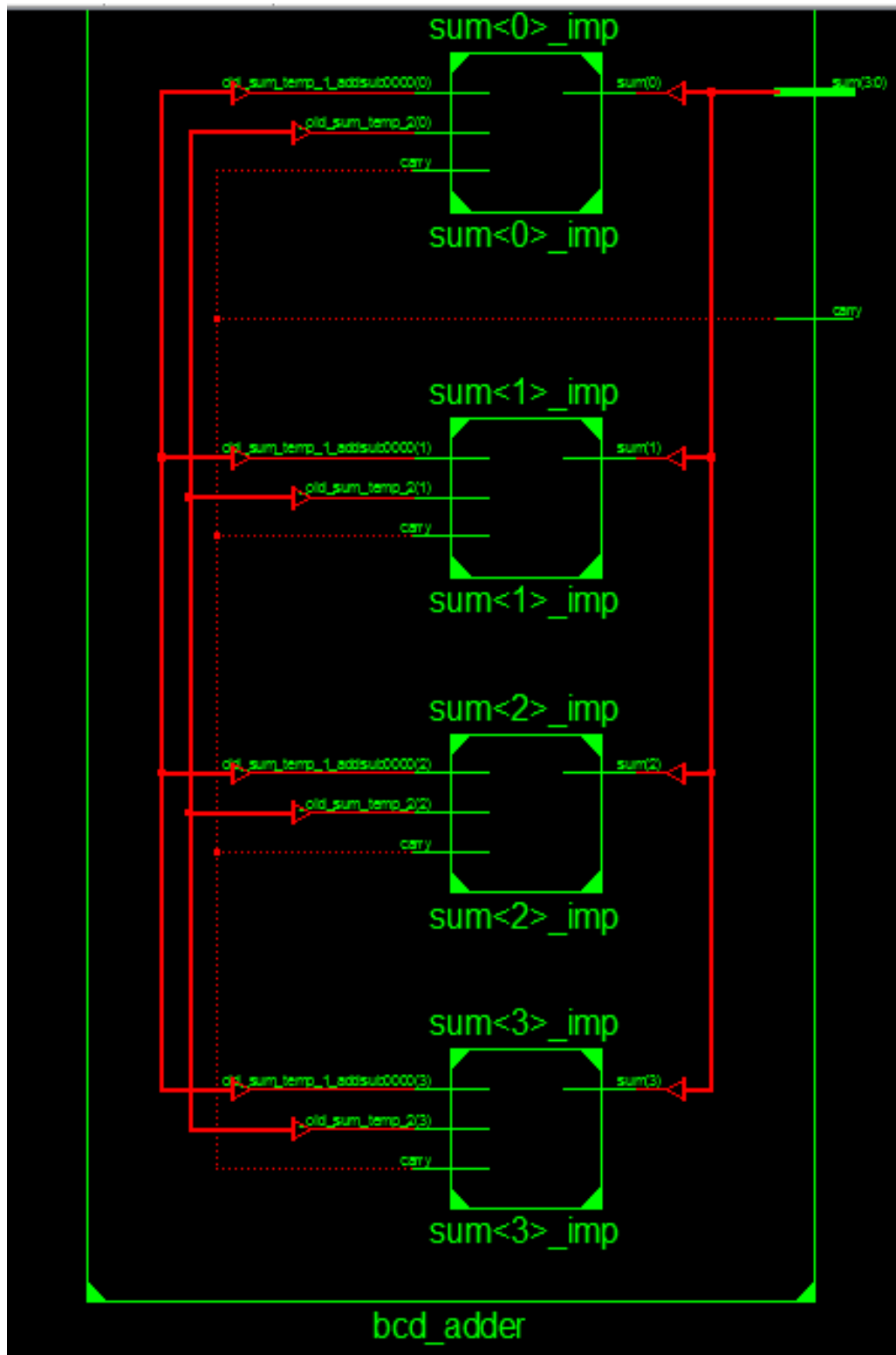


Figure 3.3: Schematic diagram of BCD adder

3.2.3 RESULTS

In table 3.3 shows design parameters of BCD. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

TABLE 3.3:Design parameters of BCD

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
BCD	8	4	7.284ns	1.777ns	16	14

LUT of BCD is 24, slices is 12 ,delay 12.67ns and cell usage 28.

3.2 QSD ADDER

Binary signed digit (BSD) number is used in digital system. Binary has two logic levels 0 and 1. This shows the high and low logic level of binary. Here 1 represent high logic level and 0 represent low logic level. Binary operation is limited due to carry generation that mean it need more interconnects in VLSI which occupy large area and increase the complexity of the circuit. Generation of carry in binary arithmetic operations occupy large area which reduce the speed of the microcomputers. To overcome these problems of binary adder we are using Quaternary Signed Digit number System (QSD).

3.3.1 QSD ADDER DESIGN

QSD adder gives carry free addition. There are mainly two steps to describe the QSD addition as shown in figure 3.4(n digit QSD adder). First step generates carry/sum and second step add the Intermediate Carry and Intermediate Sum of first step result. The steps of QSD addition briefly describe in further section 3.3.2.

B_{n-1}

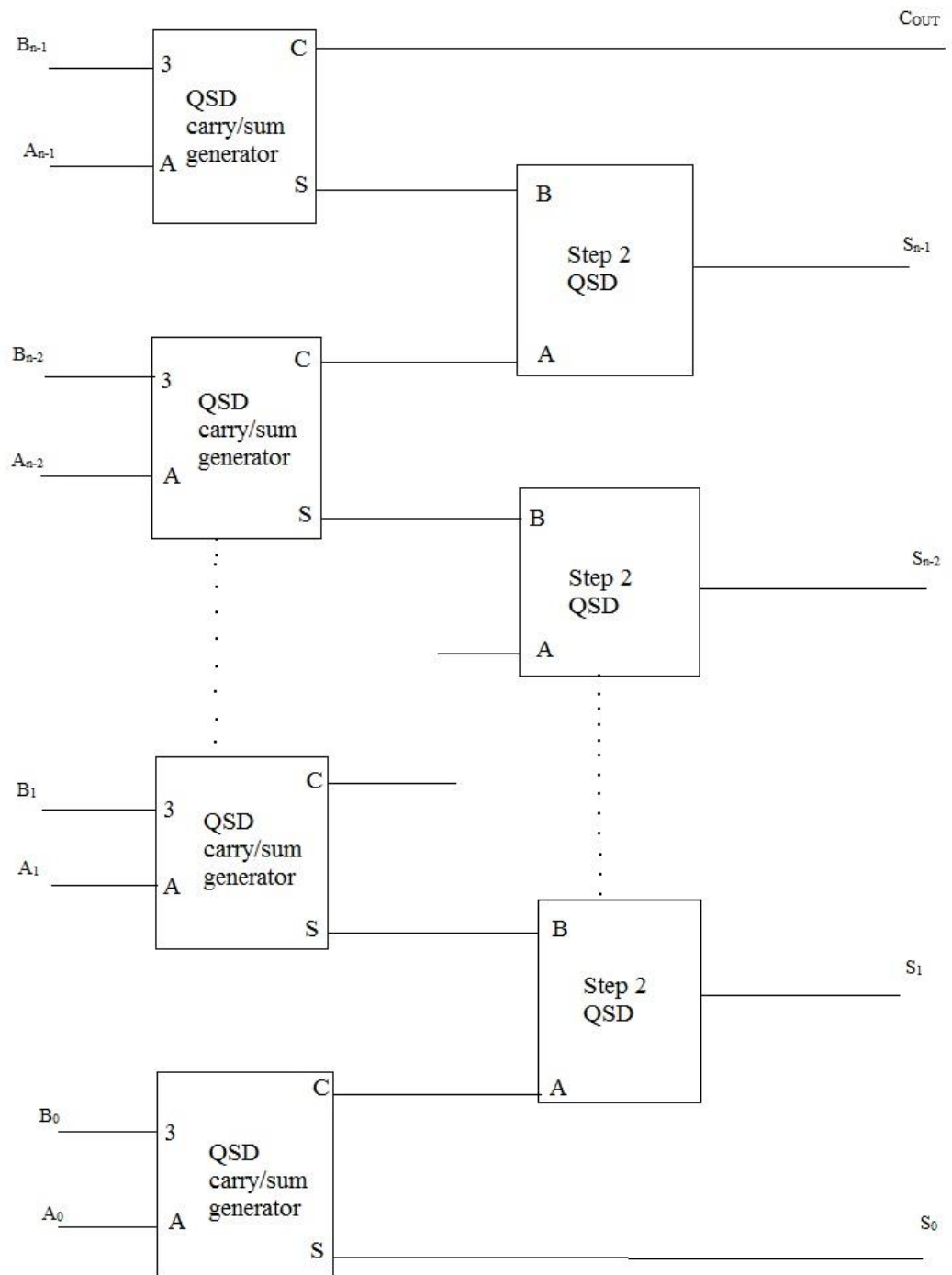


Figure 3.4: n digit QSD adder [3].

3.3.2 DESIGN ALGORITHM

In this section we are studying about the steps and rules of QSD adder. The maximum value of addition of two QSD number is +6 and minimum value is -6. QSD is represented by set of numbers -3,-2,-1,0, 1, 2 and 3. If we add the two maximum values from the set of QSD number the output value is 6. If we add the two minimum values from set then output result is -6. In TABLE 3.4 shown all possible maximum and minimum values of QSD addition in between (-6 to +6).

Table 3.4: Possible maximum and minimum values of addition[3]

	3	2	1	0	-1	-2	-3
3	+6	+5	+4	+3	+2	+1	0
2	+5	+4	+3	+2	+1	0	-1
1	+4	+3	+2	+1	0	-1	-2
0	+3	+2	+1	0	-1	-2	-3
-1	+2	+1	0	-1	-2	-3	-4
-2	+1	0	-1	-2	-3	-4	-5
-3	0	-1	-2	-3	-4	-5	-6

Now we discuss how QSD addition will work, most of the arithmetic operation deals in decimal number or base 10. The first step of the QSD addition is to convert decimal number into QSD number as shown in figure 3.5.

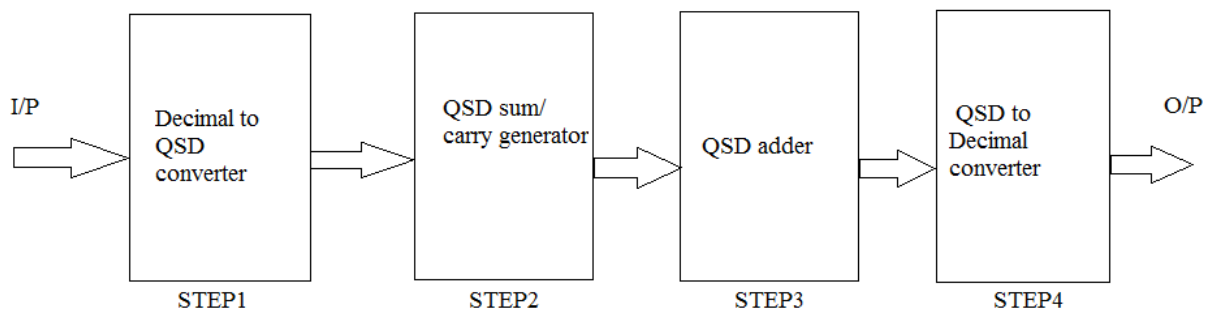



Figure 3.5: Basic concept of QSD addition[6].

Let's take an example for better understanding the steps of QSD addition. Add two unsigned decimal numbers $(102)_{10}$ and $(220)_{10}$ using QSD number.


Step 1: In step 1 we convert decimal number into QSD number.

$(102)_{10}$

4	102	2	
4	25	1	
4	6	2	
	1		

$(1212)_4$

$(220)_{10}$

4	220	0	
4	55	3	
4	13	1	
	3		

$(3130)_4$

After conversion QSD representation of decimal number $(102)_{10}$ and $(220)_{10}$ is shown in equation 1 and 2

$$(102)_{10} = (1212)_4 \quad (1)$$

$$(220)_{10} = (3130)_4 \quad (2)$$

Step 2: In this step we have two QSD numbers, we add addend and augends. After addition we can get a sum which may or may not be a QSD number. If number is QSD then it will directly converted from QSD into decimal number. Then it is not necessary to go through from step 3. If in result any number at place of one's, ten's, hundred's and so on is greater than 3 then number is not QSD. Then we move on 3rd step. Let's add two QSD numbers

$$\begin{array}{r}
 A \quad = \quad 1\ 2\ 1\ 2 \\
 B \quad = \quad 3\ 1\ 3\ 0 \\
 \hline
 \text{SUM} \quad = \quad 4\ 3\ 4\ 2 \\
 \hline
 \end{array}$$

Our target is to design a circuit which adds two QSD numbers. As we discussed if we get number greater than 3 at any place of sum then it is not a QSD number. In result we are finding that at ten's and thousand's place number is greater than 3 so it is not a QSD number. So we move on 3rd step of basic concept of QSD addition.

Table 3.5: All possible addition combination of two QSD number[13]

INPUT		OUTPUT				
QSD	BINARY	DECIMAL	QSD	BINARY		
A_i B_i	A_i B_i	SUM	IC IS	C_i S_i		
3 3	011 011	6	1 2	01 010		
3 2	011 010	5	1 1	01 001		
2 3	010 011	5	1 1	01 001		
3 1	011 001	4	1 0	01 000		
1 3	001 011	4	1 0	01 000		
2 2	010 010	4	1 0	01 000		
1 2	001 010	3	1 -1	01 111		
2 1	010 001	3	1 -1	01 111		

3 0	011 000	3	1 -1	01 111
0 3	000 011	3	1 -1	01 111
1 1	001 001	2	0 2	00 010
0 2	000 010	2	0 2	00 010
2 0	010 000	2	0 2	00 010
3 -1	011 111	2	0 2	00 010
-1 3	111 011	2	0 2	00 010
0 1	000 001	1	0 1	00 001
1 0	001 000	1	0 1	00 001
2 -1	010 111	1	0 1	00 001
-1 2	111 010	1	0 1	00 001
3 -2	011 110	1	0 1	00 001
-2 3	110 011	1	0 1	00 001
0 0	000 000	0	0 0	00 000
1 -1	001 111	0	0 0	00 000
-1 1	111 001	0	0 0	00 000
2 -2	010 110	0	0 0	00 000
-2 2	110 010	0	0 0	00 000
-3 3	101 011	0	0 0	00 000
3 -3	011 101	0	0 0	00 000
0 -1	000 111	-1	0 -1	00 111
-1 0	111 000	-1	0 -1	00 111
-2 1	110 001	-1	0 -1	00 111
1 -2	001 110	-1	0 -1	00 111

-3	2	101	010	-1	0	-1	00	111
2	-3	010	101	-1	0	-1	00	111
-1	-1	111	111	-2	0	-2	00	110
0	-2	000	110	-2	0	-2	00	110
-2	0	110	000	-2	0	-2	00	110
-3	1	101	001	-2	0	-2	00	110
1	-3	001	101	-2	0	-2	00	110
-1	-2	111	110	-3	-1	1	11	001
-2	-1	110	111	-3	-1	1	11	001
-3	0	101	000	-3	-1	1	11	001
0	-3	000	101	-3	-1	1	11	001
-1	-3	111	101	-4	-1	0	11	000
-3	-1	101	111	-4	-1	0	11	000
-2	-2	110	110	-4	-1	0	11	000
-2	-3	110	101	-5	-1	-1	11	111
-3	-2	101	110	-5	-1	-1	11	111
-3	-3	101	101	-6	-1	-2	11	110

Addition of any QSD number, results in the range between -6 to 6 because it has the maximum range to represent any decimal number in between -3 to +3. Table 3.5 shows all possible addition combinations of two QSD numbers.

Step 3: If the sum we are getting after addition is not a QSD number. Then to eliminate the carry we add Intermediate Carry (IC) and Intermediate Sum (IS). Before adding IS and IC we have to follow two rules given below.

Rule 1: While doing the addition the value of the IC should be in the range of -1 to +1.

Rule 2: The value of the IS should be in the range of -2 to +2.

Values of IC and IS evaluated from TABLE 3.6 which gives QSD code number of non QSD numbers.

Table 3.6: Possible values of IC and IS [13].

SUM	QSD CODE NUMBER		QSD ALSO REPRESENTED BY NUMBERS
	IC	IS	
-6	$\bar{1}$	$\bar{2}$	$\bar{2}2, \bar{1}\bar{2}$
-5	$\bar{2}$	$\bar{1}$	$\bar{2}3, \bar{1}\bar{1}$
-4	$\bar{1}$	0	$\bar{1}0$
-3	$\bar{1}$	1	$\bar{1}1, 0\bar{3}$
-2	0	$\bar{2}$	$\bar{1}2, 0\bar{2}$
-1	0	$\bar{1}$	$\bar{1}3, 0\bar{1}$
0	0	0	0 0
1	0	1	01, $1\bar{3}$
2	0	2	02, $1\bar{2}$
3	1	$\bar{1}$	03, $1\bar{1}$
4	1	0	1 0
5	1	1	11, $2\bar{3}$
6	1	2	12, $2\bar{2}$

The result we get after addition is $(4342)_4$, for IC as Rule 1 said it should be in between -1 to 1. From table 7, QSD representation for 3 and 4 is IC= 1, IS = $\bar{1}$ and IC = 1, IS = 0 respectively. Where IC in both the cases is 1 while IS is different for 3 and 4. For IC, at place of 3 and 4 we write 1 because it generate carry 1 and write 0 at place of 2 while for IS we shift right and at place of 3 and 4 we write $\bar{1}$ and 0 respectively, where 2 remain the same at its place. We add IS and IC to get output result in QSD number.

$$\begin{array}{r}
 \text{IC} \quad 1\ 1\ 1\ 0 \\
 \text{IS} \quad 0\ \bar{1}\ 0\ 2 \\
 \hline
 \text{OUTPUT} \quad 1\ 1\ 0\ 0\ 2 \\
 \hline
 \end{array}$$

Output is in QSD form $(11002)_4$ now further we move on step 4.

Step 4: In step 4 we are converting QSD number into decimal number.

$$\begin{aligned}
 (11002)_4 &= 1 \times 4^4 + 1 \times 4^3 + 0 \times 4^2 + 0 \times 4^1 + 2 \times 4^0 \\
 &= 256 + 64 + 2 \\
 &= (322)_{10}
 \end{aligned}$$

So the final sum of adding two QSD number is $(11002)_4$ in QSD and $(322)_{10}$ in decimal.

3.3.3 SIMULATION

The code for QSD adder is written in Verilog. Design is simulated and analyzed using Xilinx 14.7 ISE Simulator. Figure 3.6 shows the output waveform of QSD and figure 3.7 shows the schematic diagram. In figure 3.6 we are adding two QSD numbers 3 and 2, in result we get sum 5. Sum is not a QSD number we move on step 3. In step 3 we add IC and IS. From table 3.6 IC of 5 is 1 and IS is also 1. After addition of IC and IS final sum of two QSD number is 11. Final sum we get is a QSD number.

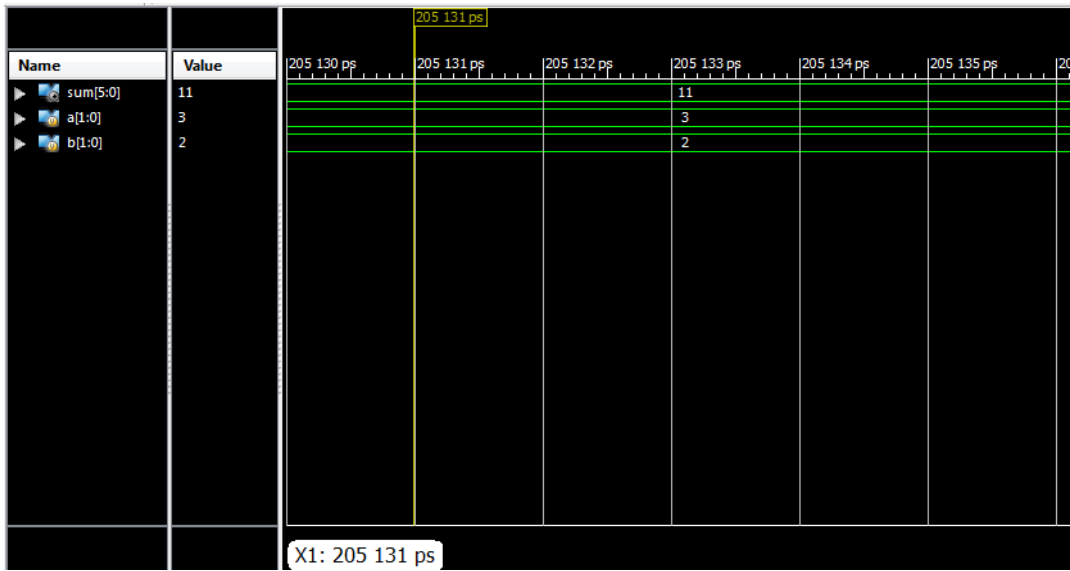


Figure 3.6: Output wave form of QSD

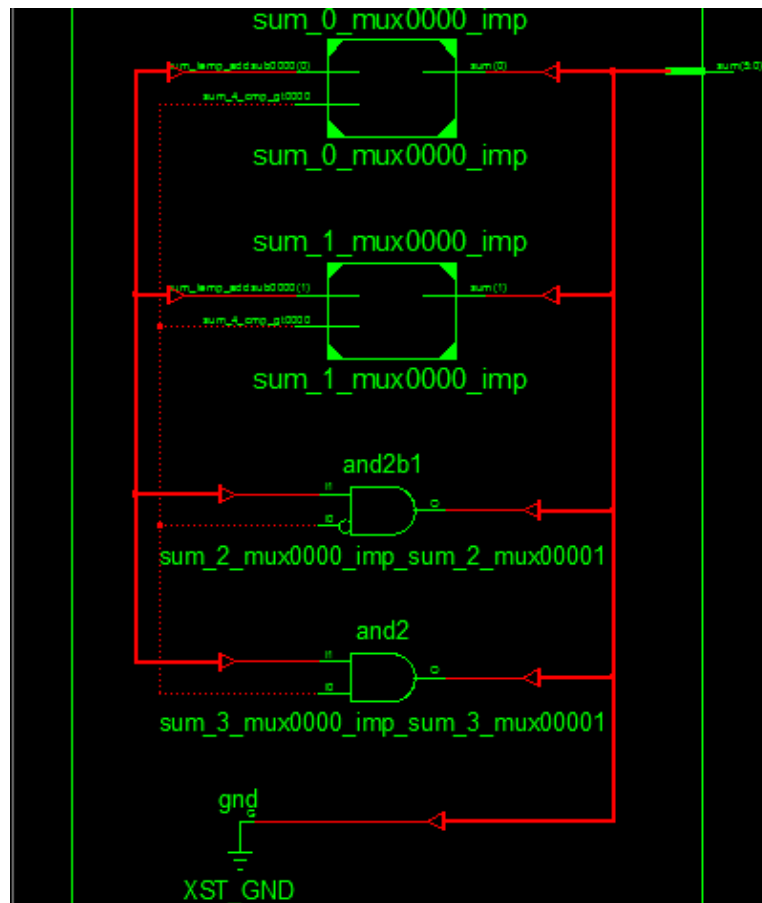


Figure 3.7: Schematic diagram of QSD adder

3.3.4 RESULTS

In table 3.7 shows design parameters of QSD. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

Table 3.7: Design parameters of QSD addition

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	3	2	5.194ns	1.26ns	4	10

LUT of QSD is 3, slices is 2 ,delay 6.454ns and cell usage 14.

The final circuit was implemented on Xilinx SPARTAN 3E-100or250 FPGA board successfully as shown in Figure 3.8.

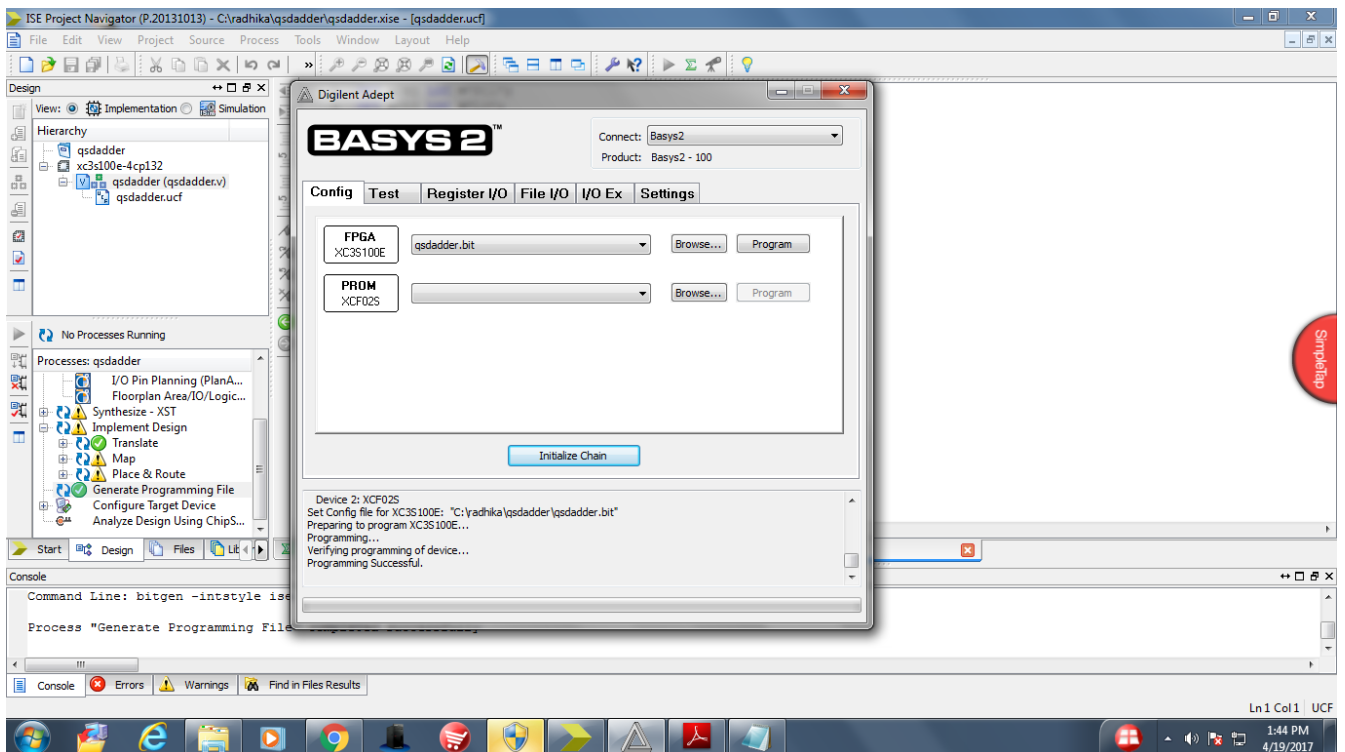


Figure 3.8: Device successfully programmed on FPGA board

3.3 QSD SUBTRACTOR

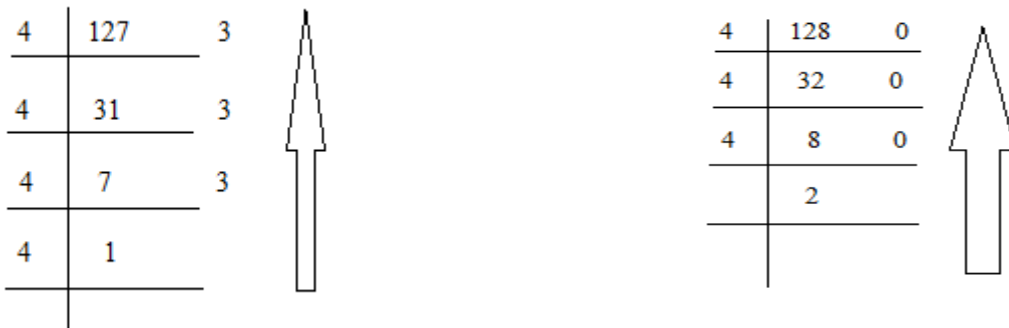
Subtraction is one of the circuits from arithmetic operations and this logic circuit is used to calculate the difference between two numbers. So in this section we implement a QSD subtractor circuit. It also has four basic steps to complete subtraction of two numbers.

3.4.1 DESIGN ALGORITHM

To understand the steps of QSD subtraction we take an example of subtraction.

Example: to perform QSD subtraction of 127 and 128.

Step 1: To convert the decimal number into QSD number



QSD number of $(127)_{10}$ is $(1333)_4$ and QSD number of $(128)_{10}$ is $(2000)_4$

Step 2: In this step we have two QSD numbers, we subtract these QSD number. After subtraction we can get a sum which may or may not be a QSD number. If number is QSD then it will directly converted from QSD into decimal number. Then it is not necessary to go through from step 3. If in result any number at place of one's, ten's, hundred's and so on is greater than 3 then number is not QSD. Then we move on 3rd step. Let's subtract two QSD numbers.

$$\begin{array}{r}
 A = 1333 \\
 B = \bar{2}\bar{0}\bar{0}\bar{0} \\
 \hline
 \text{SUM} \quad \bar{1}333 \\
 \hline
 \end{array}$$

Our target is to design a circuit which subtracts two QSD numbers. In table 3.8 ,there is all possible combination of QSD subtraction .

Table 3.8: All possible combination of two QSD subtractions [19]

INPUT				OUTPUT				
QSD		BINARY		DECIMAL	QSD		BINARY	
A	B	A	B	SUBTRACTION	D1	D0	D1	D0
-3	-3	101	101	0	0	0	000	000
-3	-2	101	110	-1	0	-1	000	111
-3	-1	101	111	-2	0	-2	000	110
-3	0	101	000	-3	-1	1	111	001
-3	1	101	001	-4	-1	0	111	000
-3	2	101	010	-5	-1	-1	111	111
-3	3	101	011	-6	-1	-2	111	110
-2	-3	110	101	1	0	1	000	001
-2	-2	110	110	0	0	0	000	000
-2	-1	110	111	-1	0	-1	000	111
-2	0	110	000	-2	0	-2	000	110
-2	1	110	001	-3	-1	1	111	001
-2	2	110	010	-4	-1	0	111	000
-2	3	110	011	-5	-1	-1	111	111
-1	-3	111	101	2	0	2	000	010
-1	-2	111	110	1	0	1	000	001
-1	-1	111	111	0	0	0	000	000
-1	0	111	000	-1	0	-1	000	111
-1	1	111	001	-2	0	-2	000	010
-1	2	111	010	-3	-1	1	111	001
-1	3	111	011	-4	-1	0	111	000
0	-3	000	101	3	1	-1	001	111
0	-2	000	110	2	0	2	000	010

0	-1	000	111	1	0	1	000	001
0	0	000	000	0	0	0	000	000
0	1	000	001	-1	0	-1	000	111
0	2	000	010	-2	0	-2	000	110
0	3	000	011	-3	-1	1	111	001
1	-3	001	101	4	1	0	001	000
1	-2	001	110	3	1	-1	001	111
1	-1	001	111	2	0	2	000	010
1	0	001	000	1	0	1	000	001
1	1	001	001	0	0	0	000	000
1	2	001	010	-1	0	-1	000	111
1	3	001	011	-2	0	-2	000	110
2	-3	010	101	5	1	1	001	001
2	-2	010	110	4	1	0	001	000
2	-1	010	111	3	1	-1	001	111
2	0	010	000	2	0	2	000	010
2	1	010	001	1	0	1	000	001
2	2	010	010	0	0	0	000	000
2	3	010	011	-1	0	-1	000	111
3	-3	011	101	6	1	2	001	010
3	-2	011	110	5	1	1	001	001
3	-1	011	111	4	1	0	001	000
3	0	011	000	3	1	-1	001	111
3	1	011	001	2	0	2	000	010
3	2	011	010	1	0	1	000	001
3	3	011	011	0	0	0	000	000

We get sum of $(\bar{1}333)_4$ which is a QSD number by definition, so we don't need step 3. But we have one rule also that IS is not greater than -2 and 2 so we are moving to step 3 even we are getting a QSD number .

Step 3: If the sum we are getting not satisfied the rules and conditions of QSD arithmetic that described below. Then to eliminate the carry we add Intermediate Carry (IC) and Intermediate Sum (IS). Before adding IS and IC we have to follow two rules given below.

Rule 1: While doing the subtraction the value of the IC should be in the range of -1 to +1.

Rule 2: The value of the IS should be in the range of -2 to +2.

$$\begin{array}{r}
 \text{IC} \quad = \quad 0 \ 1 \ 1 \ 1 \\
 \\
 \text{IS} \quad = \quad \bar{1} \ \bar{1} \ \bar{1} \ \bar{1} \\
 \hline
 \text{SUM} \quad \quad 0 \ 0 \ 0 \ \bar{1} \\
 \hline
 \end{array}$$

For 3 we have values of IC is 1 and IS is $\bar{1}$ (from table 3.6).we are writing 1 at place of 3 in IC and $\bar{1}$ at place of 3 in IS. After addition of IC and IS we get sum $(0000\bar{1})_4$. Now move on to step 4 .

Step 4: in this step we are converting QSD number into decimal number.

$$\begin{aligned}
 (0000\bar{1}) &= 0 \times 4^4 + 0 \times 4^3 + 0 \times 4^2 + 0 \times 4^1 - 1 \times 4^0 \\
 &= (-1)_{10}
 \end{aligned}$$

So the final sum of subtracting two numbers (127-128) is -1 by using QSD number system.

3.4.2 SIMULATION

The code for QSD adder is written in Verilog. Design is simulated and analyzed using Xilinx 14.7 ISE Simulator and got the results. Figure 3.9 shows the output waveform of QSD subtraction and figure 3.10 shows the schematic diagram.

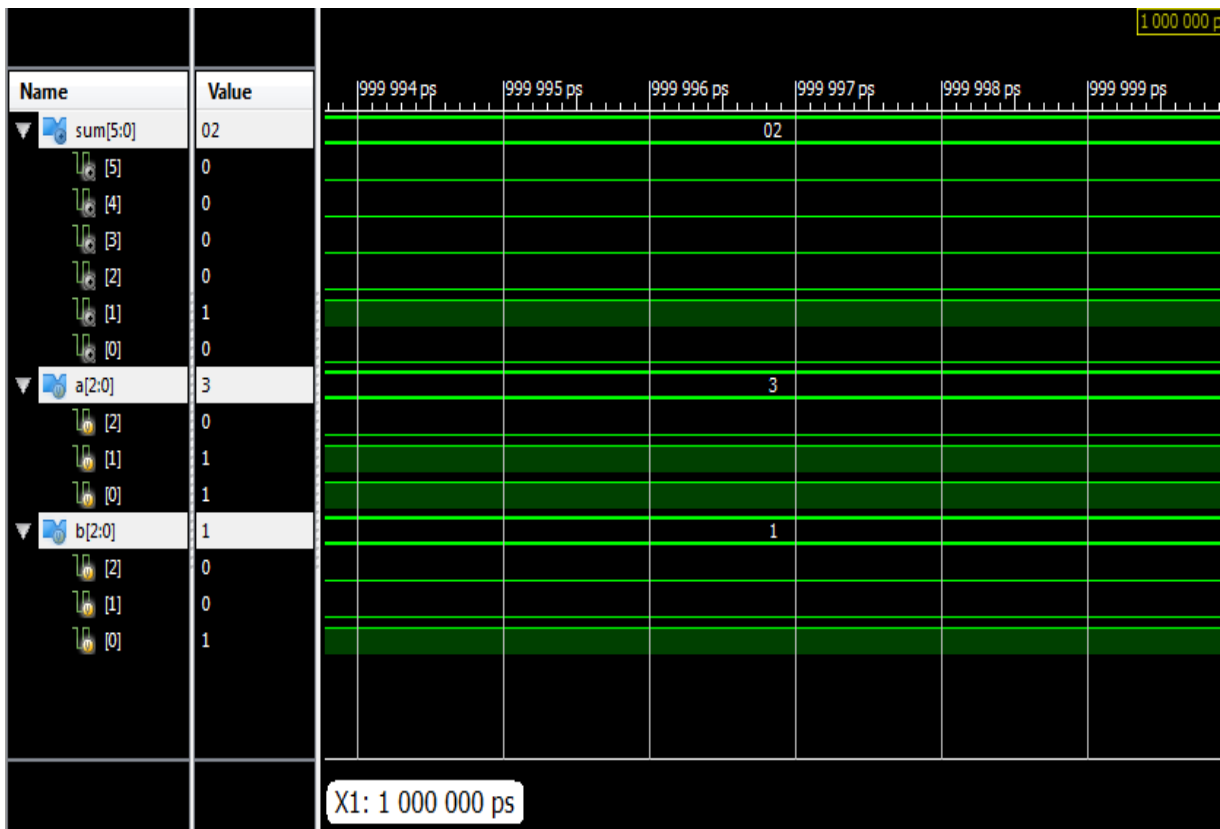


Figure 3.9: Output wave form of QSD subtraction

In figure 3.9 we are subtracting two QSD numbers 3 and 1, in result we get sum 02. Which is a QSD number we don't need to move on step 3. It fulfills the rules of subtraction rule 1 and rule 2. Rule 1 define that IC should be in between -1 to +1. Rule 2 define that IS should be in range of -2 to 2. In sum we get 2 so it fulfills all the rules of subtraction, so it skips the 3rd step.

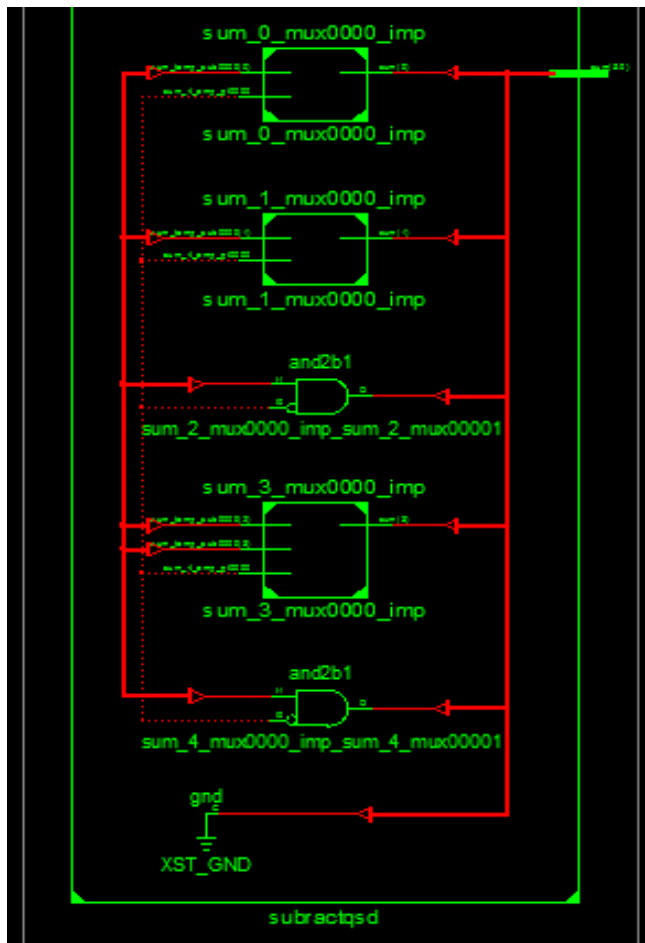


Figure 3.10: Schematic view of QSD subtraction

3.4.3 RESULTS

In table 3.9 shows design parameters of QSD. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

Table 3.9: Design parameters of QSD subtraction

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	3	2	5.194ns	1.26ns	4	10

LUT of QSD is 3, slices is 2 ,delay 6.454ns and cell usage 14.

3.4 COMPARISON

We discuss in previous section that QSD number system is a higher radix number system. To perform the arithmetic operations we use Quaternary Signed Digit Number system. QSD offers the carry free addition, borrow free subtraction and multiplication. It helps microprocessors to accomplish the task with less complexity and time delay. Now in this section we are comparing the results of BCD adder with QSD system shown in table 3.10.

Table 3.10: Comparing the results of BCD and QSD adder

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	3	2	5.194ns	1.26ns	4	10
BCD	8	4	7.284ns	1.777ns	16	14

Design summary of QSD adder gives the parameters of design look up table (LUT), slices, delay and cell usage. In Table 3.10 we have compared these design parameters of QSD with BCD circuits. LUT is which contains the output of implemented function. If LUT has “ n ” input then it can perform 2^n Boolean function. After comparison of QSD adder (shown in Table 3.10) with BCD circuits we come on conclusion that our proposed QSD multiplier uses less LUT and less no. of slices. Also it has less delay and cell usage than others. This shows that proposed QSD adder circuit is less complex than BCD adder.

CHAPTER 4

MULTIPLIERS

Multiplication is also an important part of arithmetic circuit. It consumes more area which decreases the speed of system. So it is important to design more sufficient multiplier that satisfied the design parameters (area, delay and speed). Now in this chapter we implement different types of multipliers.

4.1 WALLACE TREE MULTIPLIER

Wallace tree is an implementation of a multiplier which using 3 steps to multiply any two integers. Bit product term is generating after multiplying two integers multiplicand and multiplier in first step. In 2nd step we are using half adder and full adder to reduce the result to lower number of rows. In 3rd step using carry propagation adder two add the rows.

4.2.1 WALLACE TREE DESIGN

The method is rehashed until last stage contains just two lines. Planning a traditional Wallace tree multiplier is appeared in Fig.4.1. Design of the ordinary Wallace multiplier is done in three stages. Partial products are generated first in Wallace tree multiplier. At that point these are collected in various stages. The Wallace tree has three steps:

1. Multiply each bit of one of the arguments, by each bit of the other, yielding outcomes. Depending on position of the multiplied bits, the wires carry different weights.
2. Using half adders and full adders reduce the number of partial products into two layers.
3. Wires grouped in two numbers, and add them with adder.

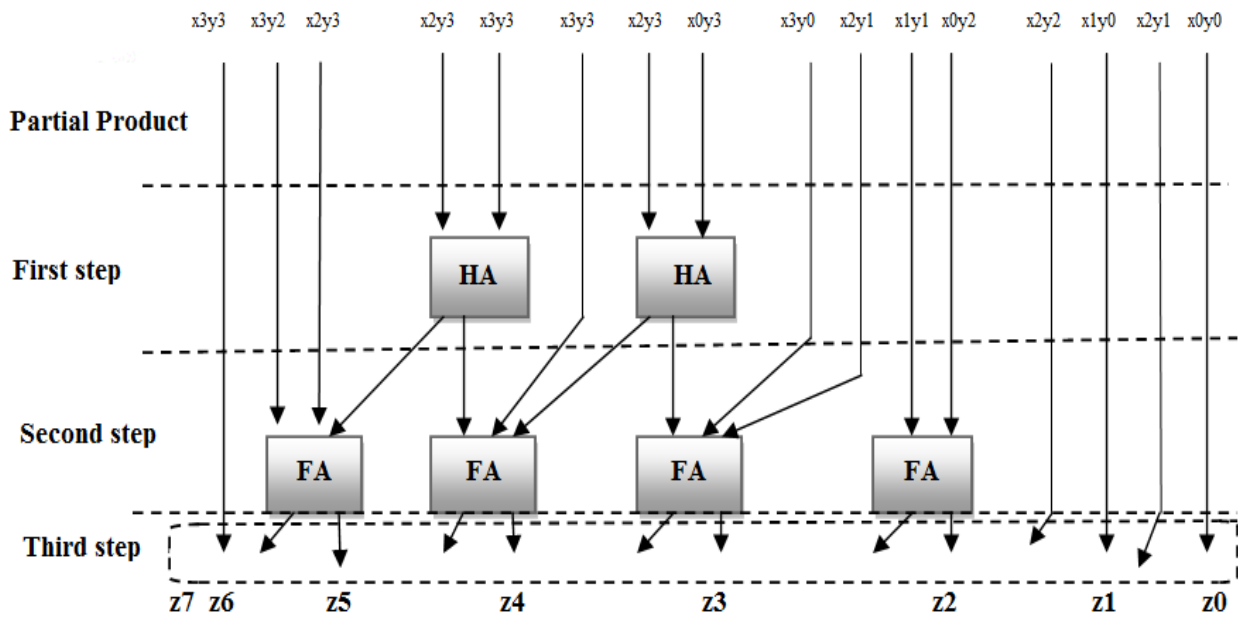


Figure 4.1: Wallace tree multiplier diagram [23].

4.2.2 SIMULATION

The code for Wallace tree multiplier is written in Verilog. Design is simulated and analyzed using Xilinx 14.7 ISE Simulator and got the results. Figure 4.2 shows the output waveform of Wallace multiplier and figure 4.3 shows the schematic diagram.

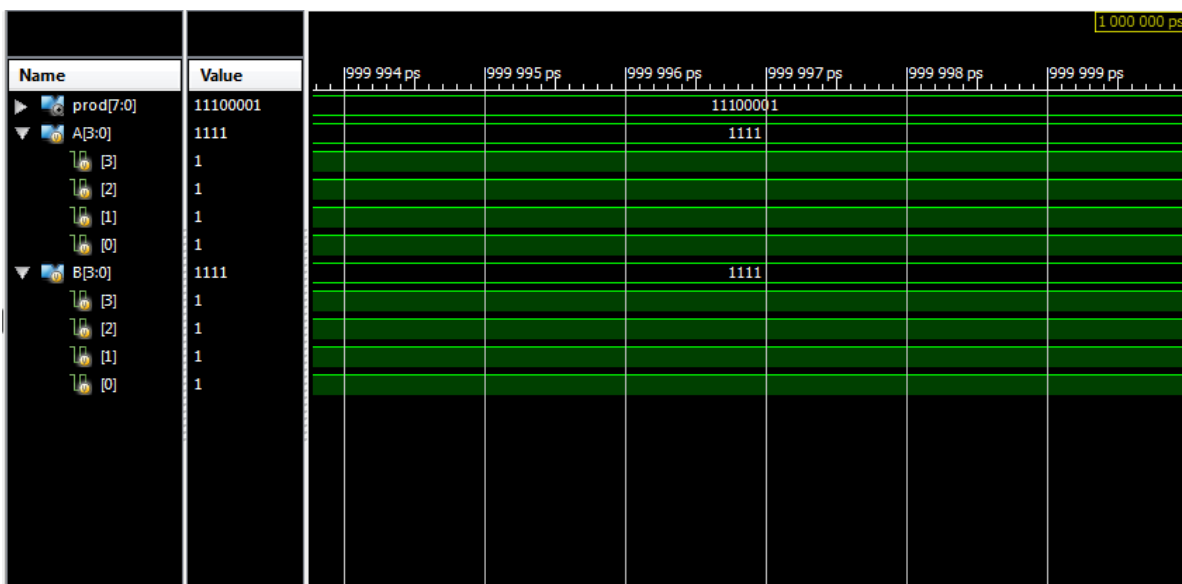


Figure 4.2: Output waveform of Wallace multiplier

We multiply two binary number 1111 and 1111, first result of multiplication is partial product . After that to reduce the complexity of the circuit we divide partial product into two layers of half adder and full adder. Addition of binary number with the help of half adder and full adder the result we get is 11100001.

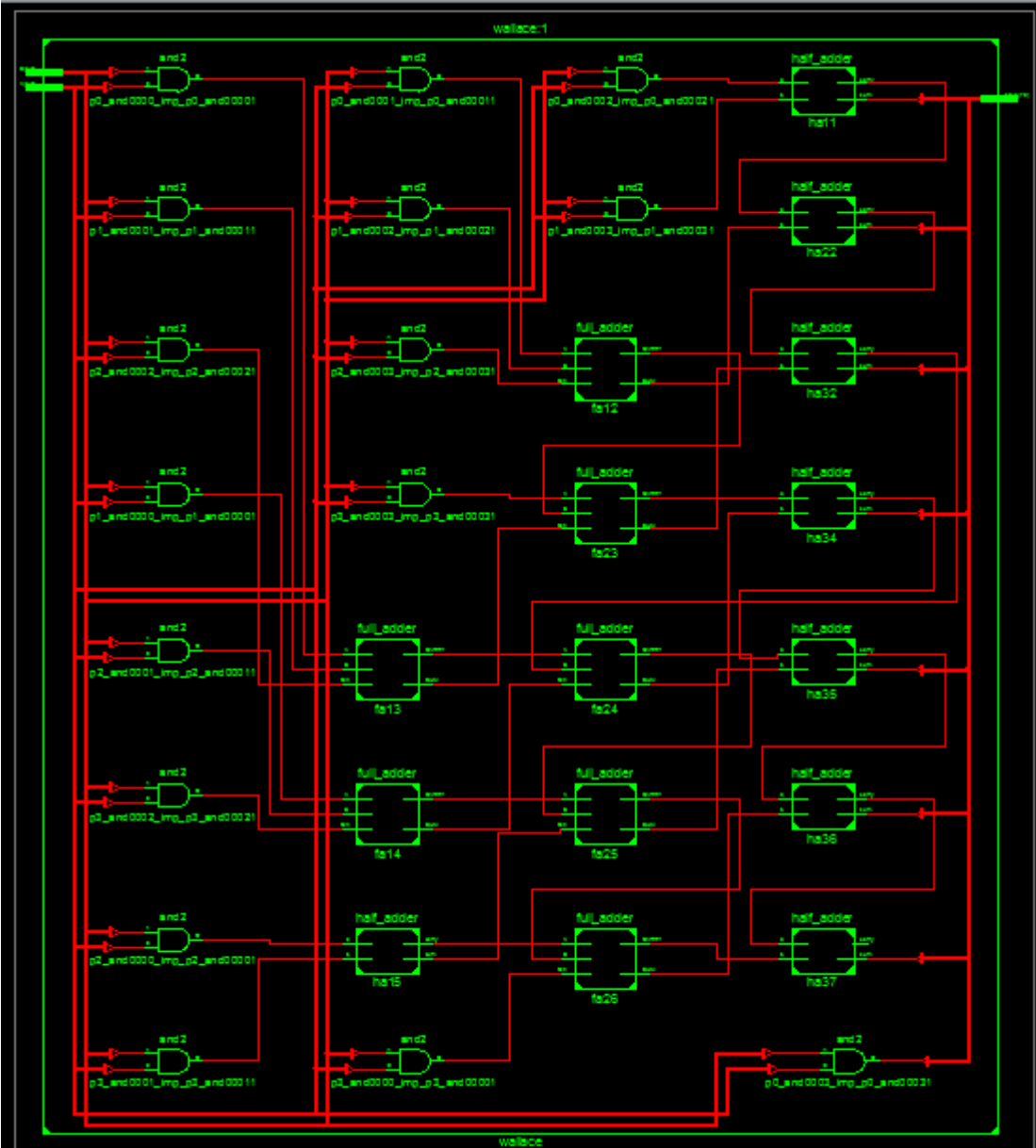


Figure 4.3: Schematic diagram of Wallace tree multiplier

4.2.3 RESULTS

In table 4.1 shows design parameters of Wallace multiplier. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

Table 4.1: Design parameters of Wallace multiplier

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	32	18	8.714ns	4.518ns	32	16

LUT of Wallace is 32, slices is 18, delay 13.232ns and cell usage 48.

4.2 BAUGH WOOLEY

Baugh–Wooley technique handles the direct multiplication of Two’s compliment numbers. Each of the partial products to be added is a signed numbers when multiplying two’s compliment numbers directly. To form a correct sum by the Carry Save Adder (CSA) tree , each partial product has to be sign extended to the width of the final product. The block diagram for 4 bit Baugh Wooley multiplier is shown in Fig 4.4

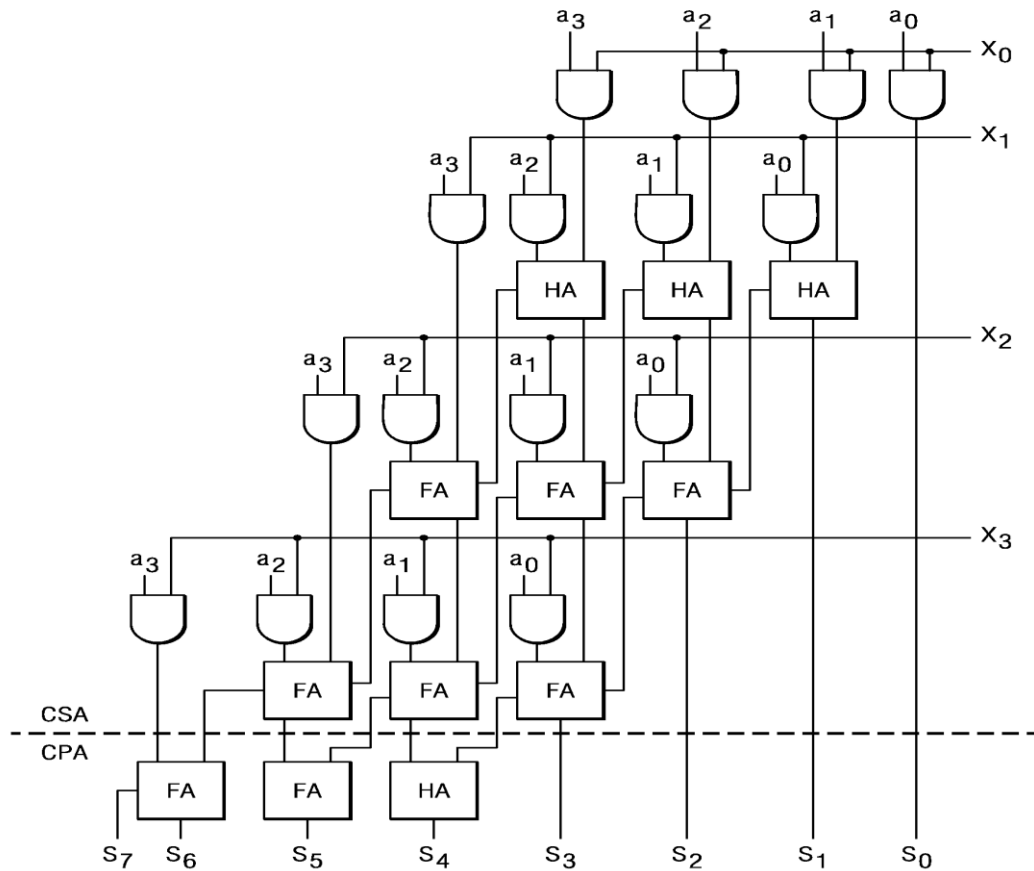


Figure 4.4: Baugh Wooley multiplier[23]

Baugh Wooley well understand by multiplying two numbers 1010 and 1111.

$$\begin{array}{r}
 1010 \text{ (this is 10 in decimal)} \\
 \times 1111 \text{ (this is 15 in decimal)} \\
 \hline
 \hline
 1010 \text{ (this is } 1010 \times 1) \\
 1010 \text{ (this is } 1010 \times 1, \text{ shifted one position to the left)} \\
 1010 \text{ (this is } 1010 \times 1, \text{ shifted two positions to the left)} \\
 + 1010 \text{ (this is } 101 \times 1, \text{ shifted three positions to the left)} \\
 \hline
 \hline
 10010110 \text{ (this is 150 in decimal)}
 \end{array}$$

4.3.1 SIMULATION

The code for Baugh Wooley multiplier is written in Verilog. Design is simulated and analyzed using Xilinx 14.7 ISE Simulator and got the results. Figure 4.5 shows the output waveform of Wallace multiplier and figure 4.6 shows the schematic diagram.

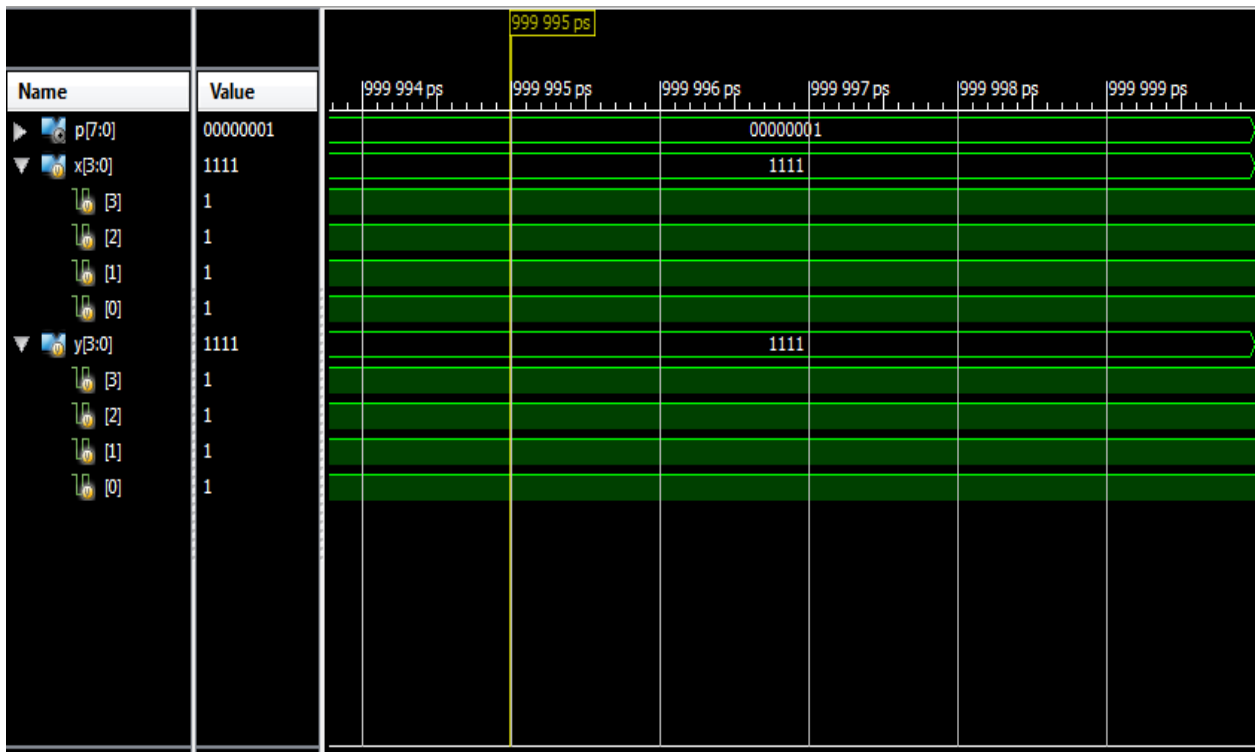


Figure 4.5: Output wave form of Baugh Wooley

In Baugh Wooley multiplier we do simple binary multiplication of two number. In each partial product we shifted one position to left at every step till the multiplication is done.

In this section we multiplying two binary number $x[3:0]=1111$ and $y[3:0]=1111$, we multiply each bit of y with x and in result we get multiplication sum is 00000001.

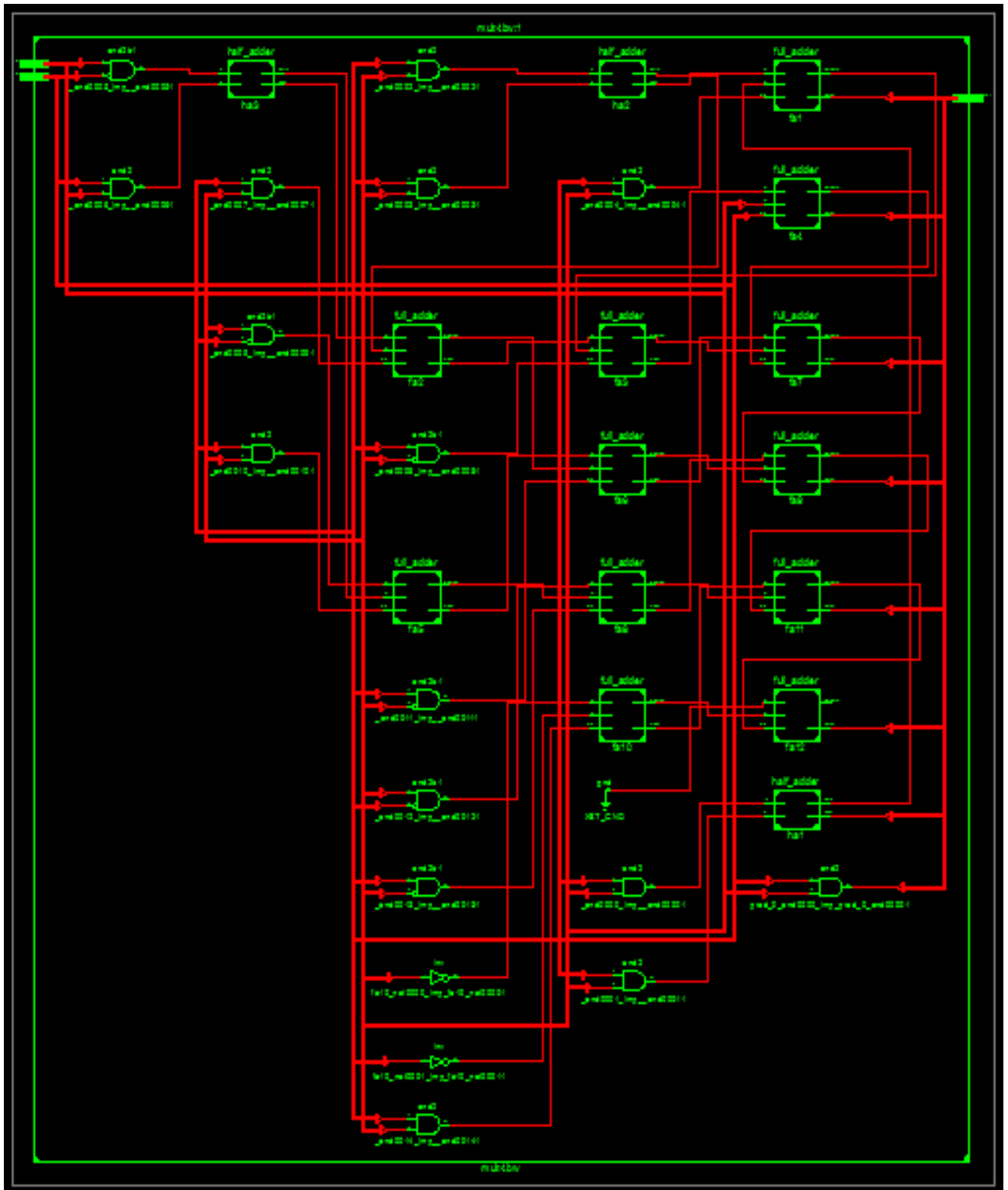


Figure 4.6: schematic daigram of Baugh Wooley

4.3.2 RESULTS

In table 4.2 shows design parameters of Baugh Wooley multiplier. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

Table 4.2: Design parameters of Baugh wooley

ADDER	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	29	17	9.418ns	4.721ns	29	16

LUT of Baugh Wooley is 29, slices is 17, delay 14.139ns and cell usage 45.

4.3 QUATERNARY SIGNED DIGIT MULTIPLIER

The binary number system for arithmetic operation generates carry which create delay and reduce the speed of microcomputers. This problem can be overcome by using higher base number system such as QSD number. It increases the speed of arithmetic operation than the binary numbers. QSD multiplier capable of carry free operations and reduces the complexity of the circuit.

4.3.1 DESIGN ALGORITHM

Our target is to design a circuit which multiplies two QSD numbers. Multiplication of any two QSD number, results in the range between -9 to 9 shown in table 4.3.

Table 4.3: Shows all possible values of multiplication of two QSD number

	-3	-2	-1	0	1	2	3
-3	9	6	3	0	-3	-6	-9
-2	6	4	2	0	-2	-4	-6
-1	3	2	1	0	-1	-2	-3
0	0	0	0	0	0	0	0
1	-3	-2	-1	0	1	2	3
2	-6	-4	-2	0	2	4	6
3	-9	-6	-3	0	3	6	9

Till now we have explained the QSD number system now we explain how we can multiply two QSD numbers. Fig 18 explains the steps of QSD multiplication in form of flow chart. There are four basic steps which are involved in the carry-free multiplication. Let's take an example to show all the possible steps of multiplication.

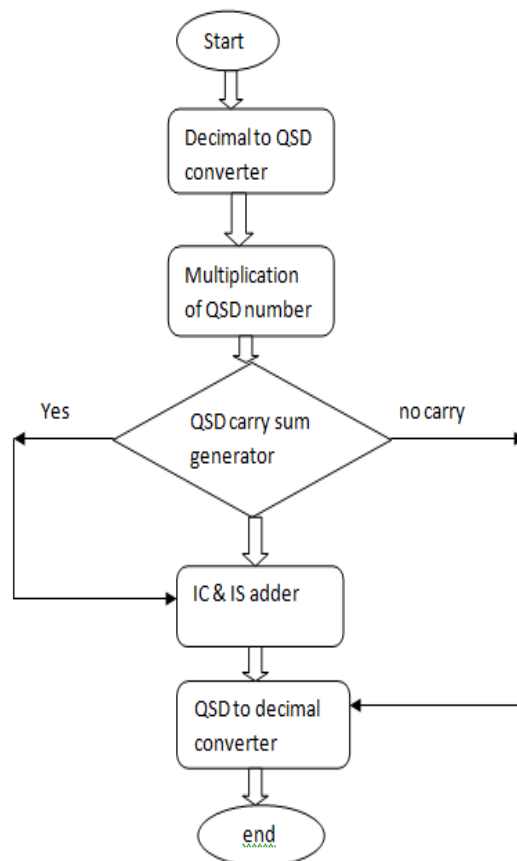


Figure 4.7: Flow chart of QSD multiplication

Step 1:- The first step of QSD multiplication is converting decimal number into QSD number.



QSD representation of decimal number $(6)_{10}$ and $(15)_{10}$ is shown in equation (3) and (4)

$$(6)_{10} = (12)_4$$

$$(15)_{10} = (33)_4$$

Step 2:- In step 2 we multiply two QSD numbers. In result we can get a sum which may or may not be a QSD number. If number is QSD then it will not generate any carry than we move on *step 4* which convert QSD into decimal number as shown in fig 4.7 (flow chart of basic logic design of multiplication).If number is not QSD then we move on *step 3* in which it generate Intermediate Sum (IS) and Intermediate Carry (IC).

In example we multiply two QSD numbers.

$$\begin{array}{r}
 A = 12 \\
 B = 33 \\
 \hline
 36 \\
 36 \\
 \hline
 \text{SUM} \quad 396
 \end{array}$$

After multiplying two QSD numbers : $(12)_4$ and $(33)_4$ we are getting sum as $(396)_4$ which is not a QSD number because at 1's place 6 is not QSD number and at 10's place 5 is not a QSD number. It will generate carry, so performing carry free multiplication we have to move on step 3.

Step 3:- If the sum we are getting after multiplication is not a QSD number then we add Intermediate Carry (IC) and Intermediate Sum (IS) to remove carry.

Table 4.4: Possible output combinations between -9 To +9

MULTIPLICATION	QSD CODE NUMBER		QSD CAN ALSO REPRESENTED BY NUMBERS
	IC	IS	
-9	$\bar{2}$	$\bar{1}$	$\bar{2}\bar{1}$, $3\bar{3}$
-6	$\bar{1}$	$\bar{2}$	$\bar{2}2$, $\bar{1}\bar{2}$
-5	$\bar{2}$	$\bar{1}$	$\bar{2}3$, $\bar{1}\bar{1}$
-4	$\bar{1}$	0	$\bar{1}$ 0
-3	$\bar{1}$	1	$\bar{1}1$, $0\bar{3}$
-2	0	$\bar{2}$	$\bar{1}2$, $0\bar{2}$
-1	0	$\bar{1}$	$\bar{1}3$, $0\bar{1}$
0	0	0	0 0
1	0	1	01, $1\bar{3}$
2	0	2	02, $1\bar{2}$
3	1	$\bar{1}$	03, $1\bar{1}$
4	1	0	1 0

5	1 1	11, $\overline{23}$
6	1 2	12, $\overline{22}$
9	2 1	21, $\overline{33}$

As we get sum a $(396)_4$, we remove carry to perform carry free multiplication. There are two rules for carry free multiplication:-

Rule 1: The value of IS should be in between -3 and 3

Rule 2: The value of IC should be in between -2 and 2.

The result we get after multiplication is $(396)_4$, for IC as Rule 2 said it should be in between -2 to 2. From table 4.4, QSD representaiton for 3 is IC= 1, IS = $\overline{1}$, for 9 IC=2 ,IS=1 and for 6 is IC = 1, IS = 2 respectively. For IC, at place of 3,9 and 6 we write 1 ,2 ,1because it generate carry . For IS we shift right and at place of 3, 9 and 6 we write $\overline{1}$, 1 and 2 respectively. We add IS and IC to get output result in QSD number.

$$\begin{array}{r}
 \text{IC} \quad 1 \ 2 \ 1 \\
 \text{IS} \quad \overline{1} \ 1 \ 2 \\
 \hline
 \text{OUTPUT} \quad 1 \ 1 \ 2 \ 2 \\
 \hline
 \end{array}$$

After addition the result is $(1122)_4$ the Step 3 is completed here now we move on to step 4.

Step 4:- In step 4 we convert QSD number into decimal number.

$$\begin{aligned}
 (1122)_4 &= 1 \times 4^3 + 1 \times 4^2 + 2 \times 4^1 + 2 \times 4^0 \\
 &= 64 + 16 + 8 + 2 \\
 &= (90)_{10}
 \end{aligned}$$

4.3.2 SIMULATION

The code for QSD multiplier is written in Verilog. Design is simulated and analyzed using Xilinx 14.7 ISE Simulator and got the results. Figure 4.8 shows the output waveform of Wallace multiplier and figure 4.9 shows the schematic diagram. In QSD multiplication we multiply two QSD number 2 and 2 in result we get sum of 4 which is not a QSD number so we move on to step 3 in which we add IC and IS . IC of 4 is 1 and IS is 0 so after addition we get final sum is 10. Final sum that we get is a QSD number.

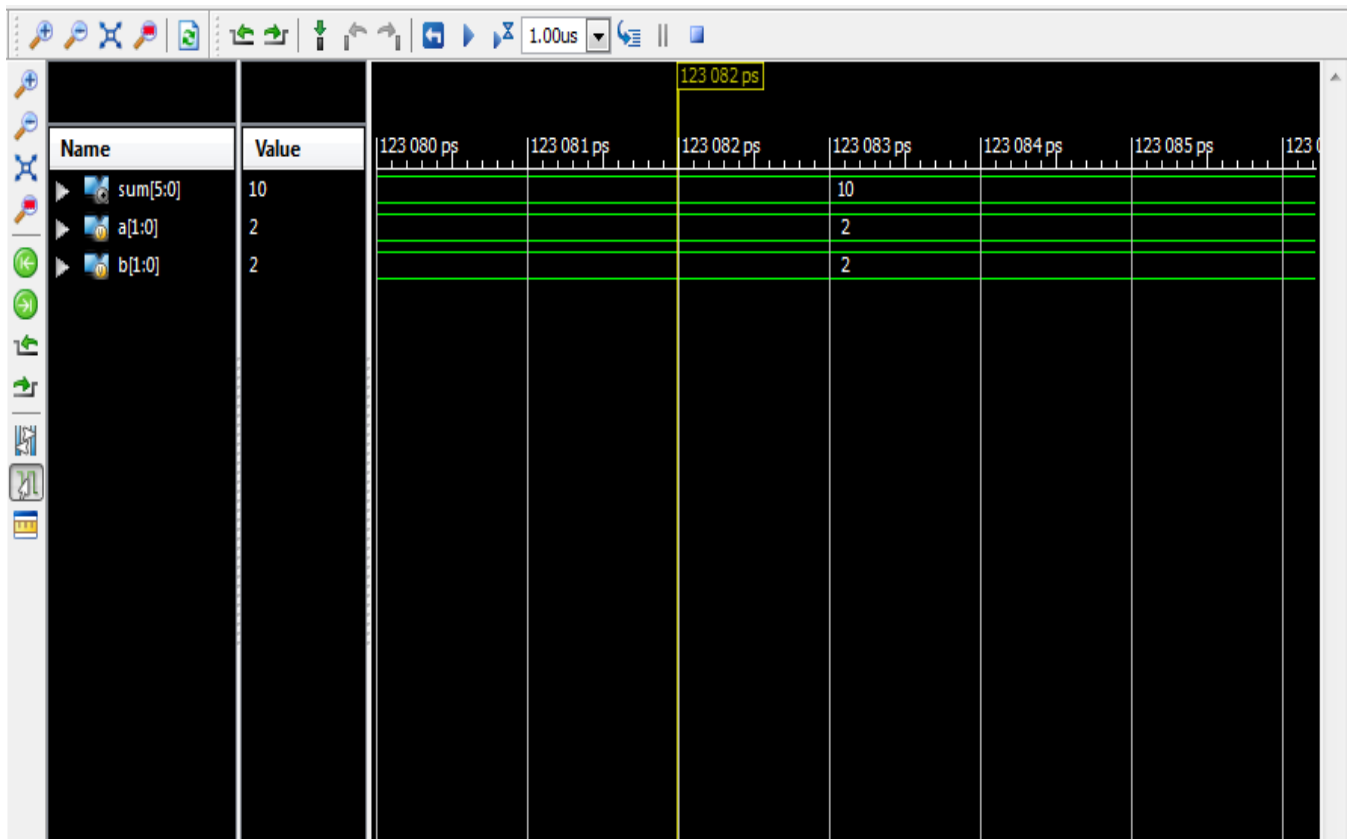


FIGURE 4.8: Output waveform of QSD multiplier

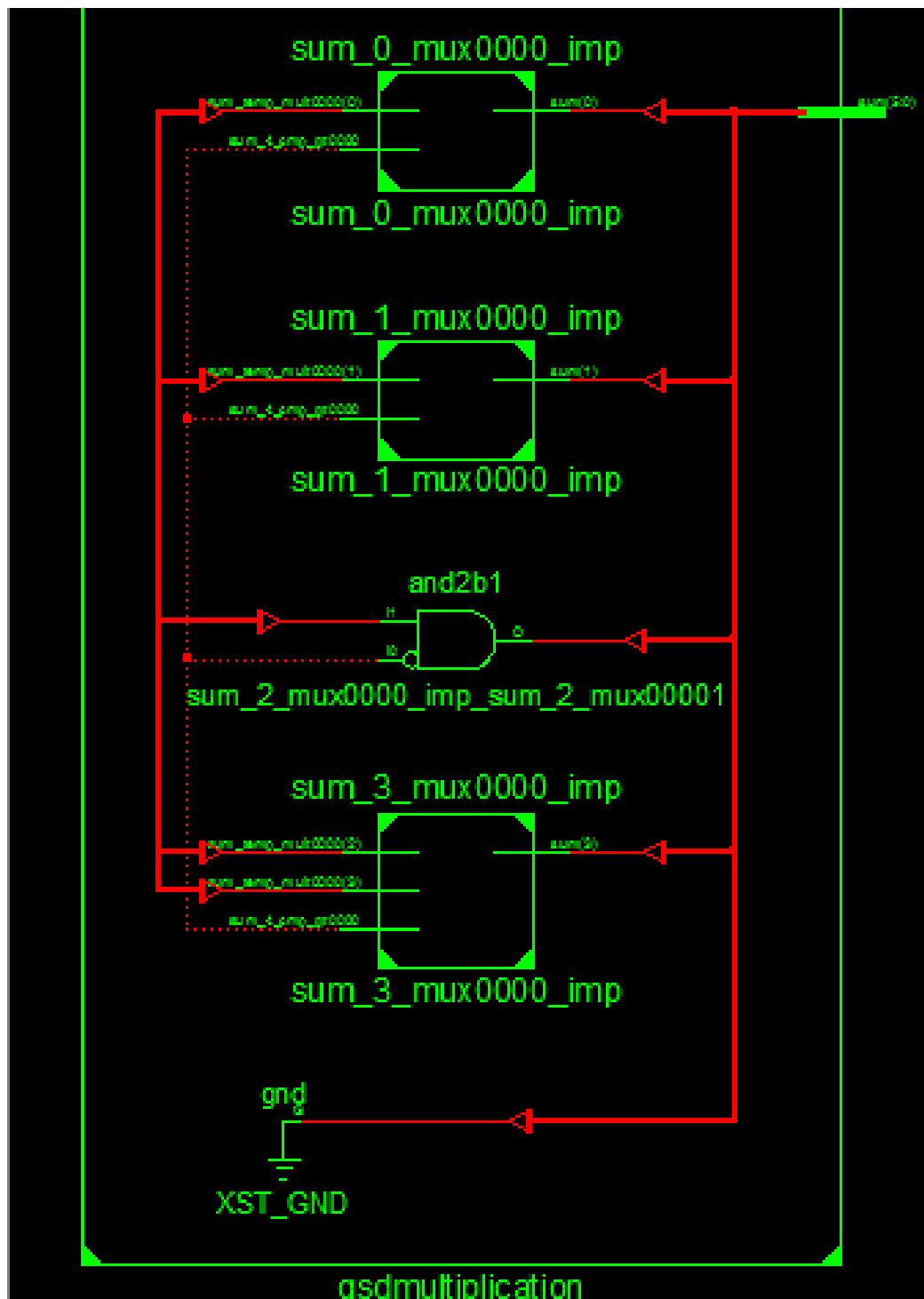


Figure 4.9: Schematic view of QSD multiplier

4.3.3 RESULTS

In table 4.5 shows design parameters of Wallace multiplier. LUT is a look up table that contains array. Look up table work is to replace runtime computation with array index operation.

Table 4.5: design parameter of QSD multiplier

Multiplier	LUT	SLICES	DELAY		CELL USAGE	
			LOGIC	ROUTE	BELS	IO BUFFERS
QSD	3	2	5.194ns	1.26ns	4	10

LUT of QSD is 3, slices is 2 ,delay 6.454ns and cell usage 14.

The final circuit was implemented on Xilinx 3E-100CP132 SPARTAN FPGA board successfully as shown in Figure 4.10.

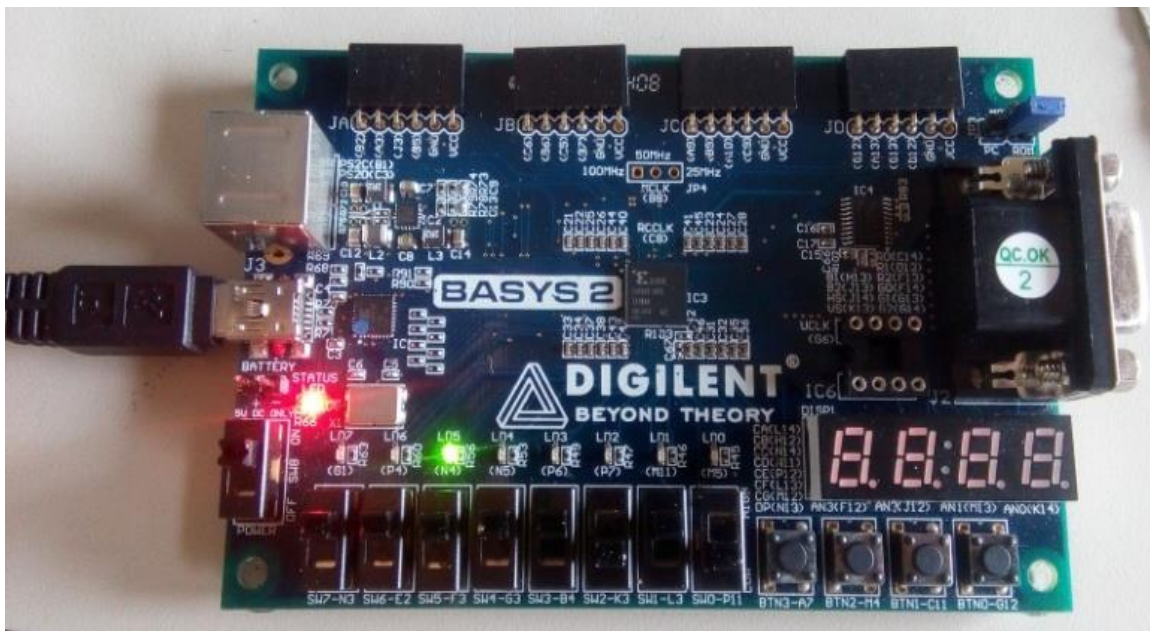


Figure 4.10: Device successfully programmed on FPGA board

4.4 COMPARISION

Now we are comparing the result of QSD multiplier with different multipliers. Table 4.6 shows the comparison of QSD multiplier with other multiplier.

Table 4.6: Comparison of QSD multiplier with different multipliers

MULTIPLIERS	LUT	NO. OF SLICES	DELAY		CELL USAGE
			LOGIC	ROUTE	
WALLACE	32	18	8.714ns	4.518ns	32
BAUGH WOOLEY	29	17	9.418ns	4.721ns	29
Proposed QSD MULTIPLIER	3	2	5.194ns	1.26ns	4

Quaternary is overcome the problem of binary arithmetic. Binary operation is limited due to carry generation that mean it need more interconnects in VLSI which occupy large area and increase the complexity of the circuit. In VLSI 70% of area is dedicated to interconnects, 20% dedicated to insulation and the remaining 10% is for device. So QSD circuit occupies less area and increase the speed of arithmetic operations. After comparison of QSD multiplier (shown in Table 4.7) with other multiplier circuits we come on conclusion that our proposed QSD multiplier uses less LUT and less no. of slices. Also it has less delay and cell usage than others. This shows that proposed QSD multiplier circuit is less complex than other circuits. We also compare proposed design with the existing design in table 4.7.

Table 4.7: Comparison of proposed design with the existing design

MULTIPLIER	METHOD	LUT	No. of SLICES	TOTAL DELAY
QSD MULTIPLIER	EXCITING [14]	12	15	11.620ns
QSD MULTIPLIER	PROPOSED	3	2	6.454ns

Comparison of existing method of QSD multiplication with our proposed design concludes that existing method uses large number of slices and LUT. Proposed design has less delay than others. It concludes that proposed design is better than the existing design.

CONCLUSION & FUTURE WORK

We already discuss that the major problem in digital world is to reduce the area and increase the speed of operations. We can obtain these parameters by using efficient optimization techniques. Arithmetic operations are very useful in any digital processor. However, propagation time delay, and circuit complexities are the major problems in arithmetic operations. For quick results in digital processor we have to increase the speed of the operation. With the use of binary number system, generation of carry in arithmetic operations create delay problems and reduce the speed of microcomputers. This paper proposes a circuit diagram which is used for carry free addition of quaternary numbers. This circuit is less complex and implemented with less delay. We have proposed QSD adder which is better than the BCD adder. We compare the result of adders and find that QSD has less delay and using less LUT in design. We also proposed QSD multiplier which is better than Wallace and Baugh Wooley multiplier. The QSD arithmetic design was implemented in Verilog HDL. The synthesis report gives us the accurate result summary of simulation in which delay of QSD arithmetic design is 6.454ns. Design of QSD arithmetic circuit allows the fast arithmetic operations without carry. These arithmetic circuits consume less energy than the other circuits.

REFERENCES

- [1] Reddy K. C. S., Dr. Rao D.V., “*High Performance Quaternary Arithmetic Logic Unit on Programmable Logic Device*”, International Journal of Advances in Applied Science and Engineering (IJAEAS), Vol. 2,(2015).
- [2] Manasa G., Rao M.D., Miranji K., “*Design And Analysis of Fast Addition Mechanism For Integers Using Quaternary Signed Digit Number System*”, International Journal of VLSI and Embedded Systems-IJVES, Vol. 05, (2014)
- [3] Rani R., Singh L.K., Sharma N., Member IEEE, “*FPGA Implementation of Fast Adder using Quaternary Signed Digit Number System,*” International Conference on Emerging Trends in Electronic and Photonic Devices & System,(2009).
- [4] Dubey S., Rani R., kumari S., and Sharma N., “*VLSI implementation of fast addition using Quaternary Signed Digit number system*”, 2013 IEEE International Conference ON Emerging Trends in Computing Communication and Nanotechnology (ICECCN), (2013).
- [5] Krishna M.N. and Ravisekhar T., “*Fast Arithmetic Operation with QSD using Verilog HDL*”, International Journal Of Engineering Science And Innovative Technology, Vol 3.(2008).
- [6] Mohan V., Mohan K. M., “*Implementation of Quaternary Signed Adder System*”, International Journal of Research Studies in Science, Engineering and Technology, Vol. 1, (2014).
- [7] Bankar, Ameya N., Hajare S., “*Design of arithmetic circuit using Quaternary Signed Digit Number system*”, International Conference on Communication and Signal Processing, (2014).
- [8] Nagamni A.N., Nishchai S., “*Quaternary high performance arithmetic logic unit design*”, 14th Euromicro Conference on Digital System Design, 2011.
- [9] Dakhane S.A., Shah A.M., “*FPGA Implementation of Fast Arithmetic Unit Based on QSD*”, international journal of computer science and information technologies(IJCSIT), Vol. 5,(2014).
- [10] Hareesh P., Chakravarti C.K. and Rao D.T., “*Design Quaternary Multiplication Using sign Digit Number Addition*”, International Journal of Trend in Research And Development, Vol. 2,(2015).
- [11] Chattopadhyay T. and Sarkar T., “*Logical Design of Quaternary Signed Digit Conversion Circuit and its Effectuation using Operational Amplifier*” Bonfring International Journal of power system and integrated circuits Vol.2, No. 3,(2012).
- [12] Ramya E., Deepti P.R., “*VLSI implementation of fast addition using Quaternary signed digit number system*”, international journal of science engineering and technology research,” Vol.4,(2015).
- [13] Piasa S., Babu S.K., “*Design of QSD Number System for Arithmetic Operations*”, International Journal of Scientific Engineering and Technology Research Vol. 03,(2014).

- [14] Babu M.K., “*Implementation of Carry Free Arithmetic Operation by QSDN*”, National Conference On Emerging Trends in Information, Digital & Embedded System, Vol. 3, (2015).
- [15] Rajkumar G., Srinivas V., “*VLSI Implementation of Signed Multiplier using Quaternary Signed Digit Number System*”, International Journal of Scientific Engineering and Technology Research, Vol. 04, (2015).
- [16] Suneetha M., Anilkumar S. and Sivakrishna P., “*Design and Implementation of 2 digit Adder using Quaternary Signed Digit Number System*”, International Journal of Electrical, Electronics and Computer Systems (IJEECS), Vol. 2, (2014).
- [17] Shende P.Y., Kshirsagar R.V., “*Quaternary Arithmetic Logic Unit Design using VHDL*”, International Journal of Science and Research (IJSR), Vol. 3, (2013).
- [18] Kumar C.V.S., Reddy P.J.R., “*Implementation of a Fast Adder Using QSD for Signed and Unsigned Numbers*”, International Journal of Science, Engineering and Technology Research (IJSETR), Vol. 3 (2014)
- [19] Sahastrabudhey S.B., Bogawar K. M., “*Arithmetic Operation Using Quaternary System using VHDL*”, IJCSET, Vol 2, (2012).
- [20] Reddy M.R., Reddy M.R., “*Implementation of fast addition with QSD using Verilog*”, International Journal of Recent Advances in Engineering & Technology, Vol. 3, (2015).
- [21] Reddy T.R., Reddy C.V.V., “*High Speed Arithmetic Operations using Quaternary Signed Digit Number System*”, International Journal of VLSI System Design and Communication Systems Vol. 02, (2014).
- [22] Manasa G., Rao M. D., Miranji K., “*Design And Analysis Of Fast Addition Mechanism For Integers Using Quaternary Signed Digit Number System*”, International Journal of VLSI and Embedded Systems-IJVES, Vol. 5, (2014).
- [23] Mohanty P., “*An Efficient Baugh-Wooley Architecture for Signed & Unsigned Fast Multiplication*”, NIET Journal Of Engineering And Technology, Vol. 1, (2013)
- [24] Mohantini P., Rajan R., “*An Efficient Baugh-Wooley Architecture for Both Signed & Unsigned Multiplication*”, International Journal of Computer Science & Engineering Technology (IJCSET), Vol 3, No. 4 (2012)
- [25] Dhivya C., Thirupathi M., Sowmiya R., “*Design of 8×8 Wallace Multiplier using MUX Based Full Adder with Compressor*”, International Research Journal of Engineering and Technology (IRJET), Vol. 2 (2015).
- [26] Swathi A.C., Yuvraj T., Praveen J., Raghavendra R.A., “*A Proposed Wallace Tree Multiplier Using Full Adder and Half Adder*”, International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, Vol. 4 (2016).
- [27] <http://www.eeeguide.com/decimal-adder-bcd-adder/>.

LIST OF PUBLICATIONS

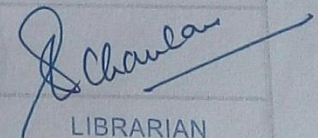
[1] Thakur R., Jain S., “ *FPGA Implementation of Adder Circuit using Quaternary Signed Digit Number System*” , pp 6273-6277, Proceedings of the 11th INDIACom: 4th 2017 International Conference on Computing for Sustainable Global Development, BVICAM, New Delhi,(2017).

[2] Thakur R., Jain S., “*FPGA Implementation of Unsigned Multiplier Circuit Based on Quaternary Signed Digit Number System*” International Journal of Engineering.

[Under review]

Submission Info

SUBMISSION ID	806380528
SUBMISSION DATE	28-Apr-2017 15:00
SUBMISSION COUNT	1
FILE NAME	M.TECH_THESIS_Radhi...
FILE SIZE	1.35M
CHARACTER COUNT	43805
WORD COUNT	9506
PAGE COUNT	67
ORIGINALITY	
OVERALL	8%
INTERNET	6%
PUBLICATIONS	5%
STUDENT PAPERS	7%
GRADEMARK	
LAST GRADED	N/A
COMMENTS	0
QUICKMARKS	


LIBRARIAN

LEARNING RESOURCE CENTRE
Jaypee University of Information Technology
Waknaghat, Distt, Solan (Himachal Pradesh)
Pin Code: 173234