ISSN No: - 2456- 2165

Low Delay Based QSD Multiplier

Pooja Bansal
M.Tech Scholar
Department of Electronics and communication
Suresh Gyan Vihar University, Jaipur, Raj, India
poojabari91@gmail.com

Vipin kumar Gupta
Assistant Professor of Electronics and communication
Suresh Gyan Vihar University
Jaipur, Raj, India
vipinsbcet@gmail.com

Abstract :- In this Paper, we are working at 4 bit QSD multiplier. In the QSD multiplier, 4 Bit QSD adder is using. QSD adder contains the addition block which is design by Reversible Logic Gate in proposed design along with pipeline by use Clock Pulse. Proposed QSD adder will use in 4 * 1 multiplier and 4 * 1 multiplier is using in 4 * 4 multiplier. As we can see from the results session, the delay is getting reduce for the proposed QSD 4-bit multiplier. Normally 4 bit, QSD multiplier is showing the delay 49.978 ns while proposed 4 Bit QSD multiplier is giving the delay of 38.919 ns. So from the results session, it is clear that proposed 4-bit QSD multiplier is working fast.

Keyword :- *Carry free addition, Fast computing, FPGA, Quaternary Signed Digit, VHDL, VLSI.*

I. INTRODUCTION

In digital systems like signal processors and computers, the arithmetic operation performs the paramount role. The system speed rises with the increasing multiplication speed. In the traditional system of binary number, a carry can propagate with entire path of the consequential digit to a large paramount. Hence the time of integration is depending on a length of word [18].

The arithmetic functions are greatly used and perform the consequential roles in the system of sundry digital like the signal processors and computers. The representation of QSD number has attracted the interest of large number of researchers. Correspondingly, the recent technological advances for ICs (Integrated Circuits) enlarge the scale of arithmetic circuits that felicitous for an implementation of VLSI. Therefore, the arithmetic operations may suffer from kenned quandaries comprising the constrained bits number and propagation time-delay, and the intricacy of circuit.

The subtraction/integration operation of QSD uses the super-tuned minter's number for any of size of operand. A multiplier is formed of adders and the partial engenderer's product. For the testing accommodation and in order for results verification, we have opted to execute all units by utilizing the contrivance of programmable logic.

The single digit (1-digit) QSD is represented by the single binary (3-bit) equipollent as below:

$$\bar{3} = 101$$

$$\bar{2} = 110$$

$$\bar{1} = 111$$

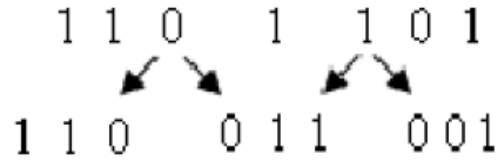
$$\bar{0} = 000$$

$$1 = 001$$

$$2 = 010$$

$$3 = 011$$

In order to convert the data of n-bit binary to the equipollent data of q-digit QSD, we need to convert certain data of n-bit binary into the binary data of 3q-bit. To accomplish a target, we need to split the 7^{th}, 5^{th} and 3^{rd} bit, that is, the aberrant bit (usually from LSB to- MSB) into the two parts. But we are not able to split a MSB. If '1' is the aberrant bit then, it's split in to 0 & 1 and if it's 0 then, it's split into the 0 & 0. The example of splitting methodology of the binary number 1101101 makes it very clear. This is shown in the below example:



Hence, we require splitting a binary data (1) q−times (for example, for the conversion of quaternary number (2-bit), a splitting is of 1-time; and for converting the quaternary number of 3-digit the split is of 2-times and many more). In every splitting, the single extra-bit is created. So, needed binary bits for the conversion are as follows:

QSD equipollent (n) = (Total numbers of bits engendered after divisions) – (extra bit engendered due to splitting) *.....(1)*

$$n = 3q - \{1 * (q-1)\}$$

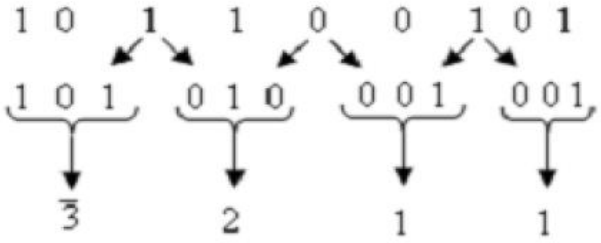
$$= (2q + 1) \quad \text{.....(2)}$$

The binary number bits will be the 9, 7, 5, and 3, and so on for converting it to an equipollent QSD-number. Currently, each 3-bit may be converted to an equipollent QSD-bit that

ISSN No: - 2456- 2165

according to an equation (3.2). The following number of examples provided will make out the things very clear.

Let, $(-155)_{10}=(101100101)_2$ have converted to the QSD equipollent $(101100101)_2$, it is of 9-bit data. Where 3rd -bit is '1', the 5th bit is '0' and the 7th -bit is '1'. Hence, from equation (3.2), we may express verbally with its QSD-equipollent is of the 4-digit. Therefore, according to a technique of splitting that expressed verbally above data binary, the data may be expressed as:



Hence, a QSD-equivalent of the number (101100101)2 is $(\bar{3}\ 2\ 1\ 1)_4$

II. QSD ADDITION

In QSD number system carry propagation chain are eliminated which reduce the computation time substantially, thus enhancing the speed of the machine. As range of QSD number is from -3 to 3, the addition result of two QSD numbers varies from -6 to +6. Table I depicts the output for all possible combinations of two numbers. The decimal numbers in the range of -3 to +3 are represented by one digit QSD number. As the decimal number exceeds from this range, more than one digit of QSD number is required. For the addition result, which is in the range of -6 to +6, two QSD digits are needed. In the two digits QSD result the LSB digit represents the sum bit and the MSB digit represents the carry bit. To prevent this carry bit to propagate from lower digit position to higher digit position QSD number representation is used. QSD numbers allow redundancy in the number representations. The same decimal number can be represented in more than one QSD representations. So such QSD represented number which prevents further rippling of carry is chosen.

A. Steps for Carry free addition

To perform carry free addition, the addition of two QSD numbers can be done in two steps [4]:

Step 1: First step generates an intermediate carry and intermediate sum from the input QSD digits i.e., addend and augend.
Step 2: Second step combines intermediate sum of current digit with the intermediate carry of the lower significant digit.

So the addition of two QSD numbers is done in two stages. First stage of adder generates intermediate carry and intermediate sum from the input digits. Second stage of adder adds the intermediate sum of current digit with the intermediate carry of lower significant digit.

This addition process can be well understood by following examples:

Sum	QSD representation	QSD coded number
-6	$\bar{2}\,2\,,\bar{1}\bar{2}$	$\bar{1}\bar{2}$
-5	$\bar{2}3,\bar{1}\bar{1}$	$\bar{1}\bar{1}$
-4	$\bar{1}0$	$\bar{1}0$
-3	$\bar{1}1,0\bar{3}$	$\bar{1}1$
-2	$\bar{1}2,0\bar{2}$	$0\bar{2}$
-1	$\bar{1}3,0\bar{1}$	$0\bar{1}$
0	00	00
1	$01,1\bar{3}$	01
2	$02,1\bar{2}$	02
3	$03,1\bar{1}$	$1\bar{1}$
4	10	10
5	$11,2\bar{3}$	11
6	$12,2\bar{3}$	12

Table I. QSD Number Representation for Carry free

To perform QSD Addition of two numbers A = 113 and B = 107 (both numbers are +'ve, and performing addition like 113+107). First convert the decimal number to their equivalent QSD representation:

$(113)_{10} = 1 \times 43 + 3 \times 42 + 0 \times 41 + 1\times 40= (1\ 3\ 0\ 1)_4$ QSD
$(107)_{10} = 1 \times 43 + 2 \times 42 + 3 \times 41 + 0\times 40= (1\ 2\ 2\ 3)_4$ QSD

Now the addition of two QSD numbers can be done as follows:

A = 113 1 3 0 1
B = 107 1 2 2 3

Decimal sum 2 5 2 4

IC 0 1 0 1
IS 2 1 2 0
sum 3 1 3 0

carry 0

The sum output is $(3\ 1\ 3\ 0)_4$ QSD which is equivalent to $(220)_{10}$ and carry output is 0. From these examples it is clear that the QSD addition design process also will carry two

ISSN No: - 2456- 2165

stages for subtraction. The QSD representations according to these rules are shown in Table 3.1 for the range of -6 to +6. As the range of intermediate carry is from -1 to +1, it can be represented in 2 bit binary number but we take the 3 bit representation for the bit compatibility with the intermediate sum. At the input side, the addend Ai is represented by 3 variable input as A2, A1, A0 and the augend Bi is represented by 3 variable input as B2, B1, B0. At the output side, the intermediate carry IC is represented by IC2, IC1, IC0 and the intermediate sum IS is represented by IS2, IS1, IS0.The six variable expressions for intermediate carry and intermediate sum in terms of inputs (A2, A1, A0, B2, B1 and B0). So we get the six output expressions for, IC2, IC1, IC0, IS2, IS1 and IS0.Using 6 variables K-map, the logic equations can be derived for the intermediate carry and intermediate sum. Using these equations block diagram can be designed.

III. MULTIPLIER DESIGN

There are generally two methods for a multiplication operation: parallel and iterative. QSD multiplication can be implemented in both ways, requiring a QSD partial product generator and QSD adder as basic components. A partial product, M_i, is a result of multiplication between an *n*-digit input, A_{n-1}-A_0, with a single digit input, B_i, where $i = 0..n-1$. The primitive component of the partial product generator is a single-digit multiplication unit whose functionality can be expressed as shown in Table II.

A \ B	-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

Table II. The outputs of all possible combinations of a pair of multiplicand (A) and multiplier (B).

The single-digit multiplication produces M as a result and C as a carry to be combined with M of the next digit. The range of both outputs, M and C, is between -2 and 2.

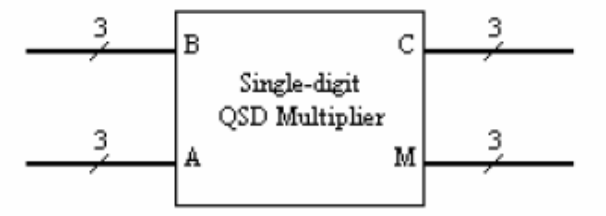


Fig 1. A single-digit QSD multiplier

The implementation of an n-digit partial product generator uses n units of the single-digit QSD multiplier. Gathering all the outputs to produce a partial product result presents a small challenge. The QSD representation of a single digit multiplication output, shown in Table III, contains a carry-out of magnitude 2 when the output is either -9 or 9. This prohibits the use of the second step QSD adder alone as a gatherer. In fact, we can use the complete QSD adder from the previous section as the gatherer. Furthermore, the intermediate carry and sum circuit can be optimized by not considering the input of magnitude 3. The QSD partial product generator implementation is shown in Figure 2.

Mult	QSD Represented Number	QSD Coding Number
-9	$\bar{2}\,\bar{1}\,,\bar{3}\,3$	$\bar{2}\,\bar{1}$
-6	$\bar{2}\,2\,,\bar{1}\,\bar{2}$	$\bar{1}\,\bar{2}$
-4	$\bar{1}\,0$	$\bar{1}0$
-3	$\bar{1}\,1\,,0\,\bar{3}$	$\bar{1}\,1$
-2	$\bar{1}\,2\,,0\,\bar{2}$	$0\,\bar{2}$
-1	$\bar{1}\,3\,,0\,\bar{1}$	$0\,\bar{1}$
0	00	00
1	$01,1\bar{3}$	01
2	$02,1\bar{2}$	02
3	$03\,,\,1\bar{1}$	$1\bar{1}$
4	10	10
6	$12,2\bar{1}$	12
9	$21,3\bar{3}$	21

Table III. The QSD representation of a single-digit multiplication output

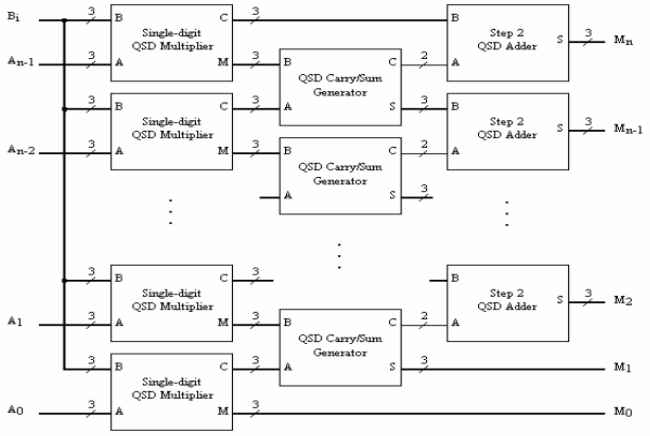


Fig 2. The *n*-digit QSD partial product generator

An n x n-digit QSD multiplication requires n partial product terms. In an iterative implementation, a 2n digit QSD adder is used to perform add-shift operations between the partial product generator and the accumulator. After n iterations, the multiplication is complete. In contrast, a parallel implementation requires n partial product circuits and n-1 QSD adder units. A binary reduction sum is applied to reduce

ISSN No: - 2456- 2165

the propagation delay to O(log n). The schematic of a 4x4 parallel QSD multiplication is shown in Figure 3.

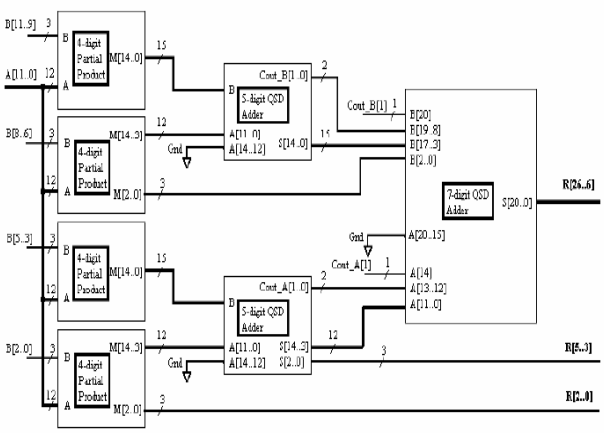


Fig 3. The 4x4 parallel QSD multiplication circuit

IV. PROPOSED METHODOLOGY

QSD is a carry free addition which is performing for increase the speed of the desire system . In the normal QSD for addition of carry and sum till to present time normal full adder is using for addition .

A. Methodology 1

We will apply Peres gate based full adder by which we can further improve the performance of the delay .We design the full adder Peres reversible logic gate based .

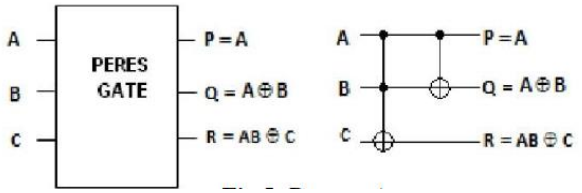


Fig 4. Peres Gate

The input vector is I (A, B, C) and the output vector is O (P, Q, R). The output is defined by P = A, Q = AB and R=AB C. Quantum cost of a Peres gate is 4. In the proposed design Peres gate is used because of its lowest quantum cost.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	1	1	0	0

Table IV. Peres gate Truth Table

A full- adder using two Peres gates is as shown in fig . The quantum realization of this shows that its quantum cost is 8 two Peres gates are used A single 4*4 reversible gate called PFAG gate with quantum cost of 8 is used to realize the multiplier.

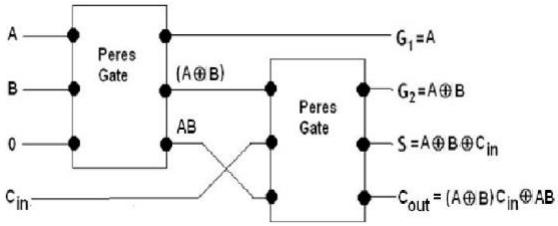


Fig 5. Full adder using two Peres gates

B. Methodlogy2

In this we apply the pipelining for further reduce the delay of the circuit . Firstly we apply pipelining in the in the QSD carry sum generator and then we apply the pipeline in the bit transfer values of carry and sum . In the pipeline we can transfer more than one clock by a single clock .

Methodology 1 and Methodology 2 is working for the QSD adder. QSD adder is using in QSD multiplication . QSD adder is using in figure 5. Figure 5 is showing the block diagram for 4*1 multiplication . In this block we apply the proposed methodology of adder. This block is used for 4*4 multiplication.

V. RESULTS

In the results session , we show the QSD 4 Bit multiplication . For design QSD 4*4 bit multiplication , we use Peres gate based Reversible Logic gate for addition . It is getting used in multiplication of 4*1. Now 4*1 multiplier is using in 4*4 multiplication. For compare the results , we show the results for QSD multiplication by simple adder and QSD multiplier by Reversible Logic gate. For execute the program , we are using Sparten 2 device family , Device name is XC2S515, Package is CS144 and speed grad is -6. Complete FPGA device Family is XC2S15-6CS144.

A. QSD 4 Bit Multiplication

In Figure 6, 4 bit QSD multiplier RTL schematic is showing. In QSD 4 bit multiplier A and B are the input of 12 bits while C is a output of 30 bit. In the output we get 30 bit of multiplication.

ISSN No: - 2456- 2165

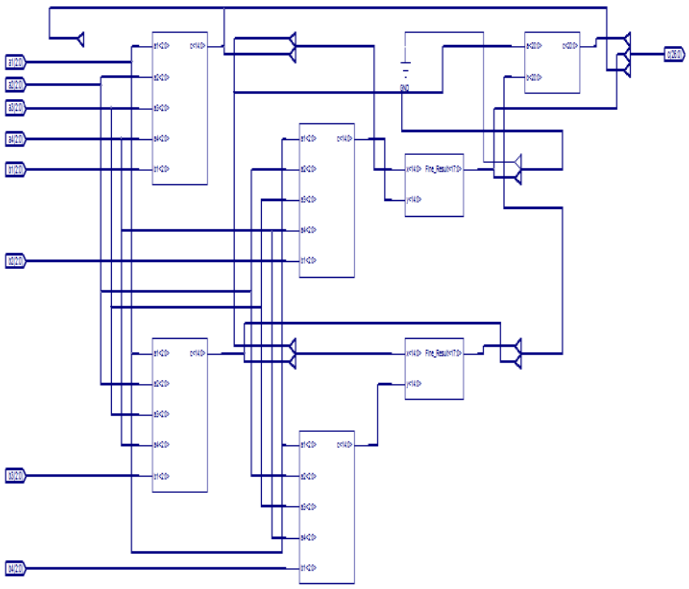


Fig 6. RTL of QSD 4 Bit Multiplication

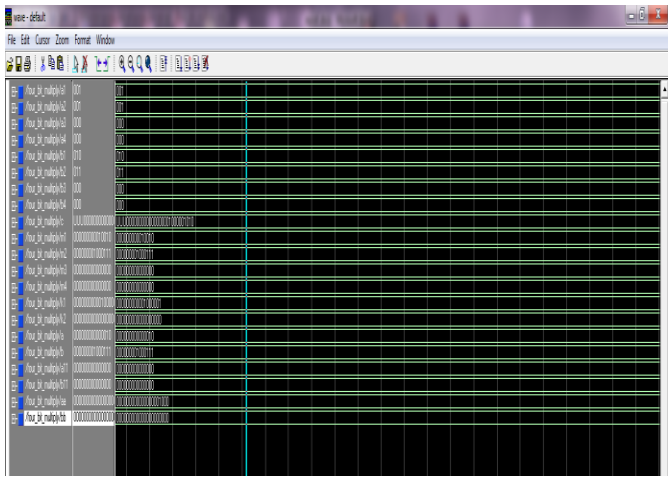


Fig 7. Output Waveform for 4 Bit QSD multiplier

Figure 7 is showing the output waveform for the QSD 4 bit multiplier. Inputs are

$$A = (5)_{10} = (0011)_4$$
$$B = (14)_{10} = (0032)_4$$

Output is coming $(1012)_4$ in QSD and $(70)_{10}$ in the decimal. QSD 4 bit multiplier is giving the delay of 49.978 ns delay for 4*4 QSD multiplier .

B. Proposed QSD 4 Bit Multiplier

In Figure 8, 4 bit QSD multiplier RTL schematic is showing. In QSD 4 bit multiplier A ,B and clock are the input. In this A and B are the input of 12 bits while C is a output of 30 bit. In the output we get 30 bit of multiplication.

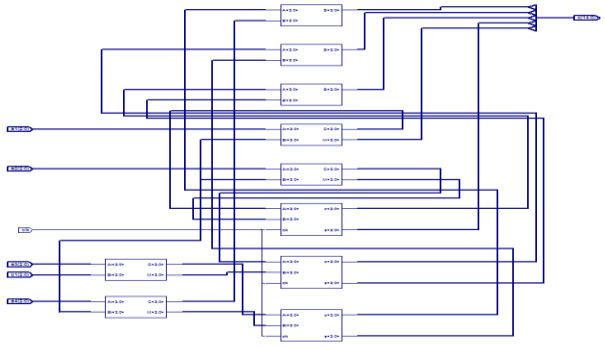


Fig 8. RTL of Propose QSD 4 Bit Multiplication

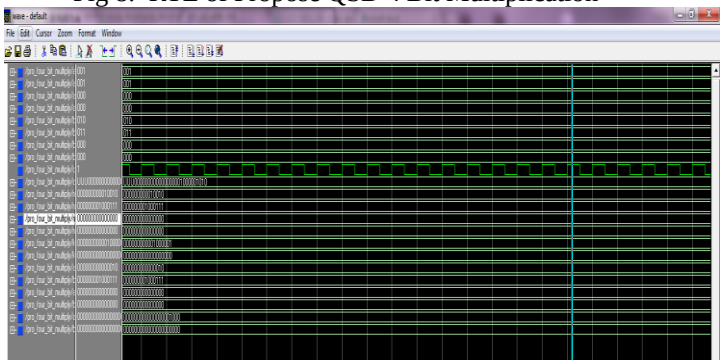


Fig 10. Output Waveform for Proposed QSD 4 bit Multiplier

Figure 10 is showing the output waveform for the QSD 4 bit multiplier. Inputs are

$$A = (5)_{10} = (0011)_4$$
$$B = (14)_{10} = (0032)_4$$

Output is coming $(1012)_4$ in QSD and $(70)_{10}$ in the decimal. QSD 4 bit multiplier is giving the delay of 38.919 ns delay for 4*4 QSD multiplier .

Design	Delay(ns)
QSD 4 Bit Multiplier	49.978 ns
Proposed QSD 4 Bit Multiplier	38.919 ns

Table V. Comparison Table for QSD 4 Bit Multiplier and Proposed QSD 4 Bit Multiplier

VI. CONCLUSION AND FUTURE SCOPE

A. Conclusion

In this Research, we are improving the performance of 4 bit QSD multiplication. As we know, QSD multiplication is fast multiplication process. But we can further increase the QSD multiplication process. For improving the speed of multiplication, we have to reduce the delay. For reduce the delay we are working at the reversible logic gate along with pipelining. We are applying these methodology in the QSD

ISSN No: - 2456- 2165

adder. QSD adder is using in the QSD multiplication. For show the delay of existing and proposed design, we are working on XC2s15-6CS 144 design. As we can see from the results session that delay is getting reduce by applying reversible logic gate. Normally, 4 Bit QSD multiplication is giving the delay of 49.978 ns while the proposed QSD 4 Bit multiplication is showing the delay of 38.919 ns.

B. Future Work

In the Future, we can further reduce delay by use some other reversible logic gate like MOG and COG. In the future, we can use MIG and COG based gate for further reduce the delay.

References

[1]. Kothuru.Ram Kumar, P. Venkata Ramana," Design And Implementation Of QSD Adders For Arithmetic Operation ", International Journal Of Professional Engineering Studies, Volume II/Issue 3/JUNE2014.

[2]. Prashant Y. Shende, Dr. R. V. Kshirsagar," Quaternary Multiplier Design Using VHDL ", International Journal Of Advanced Scientific And Technical Research, Issue 3 Volume 4, July-August 2013.

[3]. Vadathya Mohan*, K. Madan Mohan," Implementation Of Quaternary Signed Adder System ", International Journal Of Research Studies In Science, Engineering And Technology Volume 1, Issue 9, December 2014.

[4]. Ch.Vishali, V.Malleswara Rao," Quaternary Addition Using VLSI ", In IJSETR , 2014.

[5]. Ms. Priti S. Kapse1, Dr. S. L. Haridas," Design Of Quaternary Adder For High Speed Applications", International Journal Of Science, Technology & Management, Volume No 04, Special Issue No. 01, March 2015.

[6]. A. Leela Bhardwaj Reddy, V. Narayana Reddy," VLSI Implementation Of Fast Addition Subtraction And Multiplication (Unsigned) Using Quaternary Signed Digit Number System", In International Journal & Magazine Of Engineering Technology , Management And Research , Volume 2 Issue 15 , 2015.

[7]. Prashant Y. Shende*, Dr. R. V. Kshirsagar," Quaternary Adder Design Using VHDL " , International Journal Of Engineering Research And Applications (IJERA), Vol. 3, Issue 3, May-Jun 2013.

[8]. Reena Rani, Laxmi Kant Singh, Neelam Sharma," FPGA Implementation Of Fast Adders Using Quaternary Signed Digit Number System", International Conference On Emerging Trends In Electronic And Photonic Devices & Systems , 2009.

[9]. Shrikesh A. Dakhane, A. M. Shah," FPGA Implementation Of Fast Arithmetic Unit Based On QSD", International Journal Of Computer Science And Information Technologies, Vol. 5 (3) , 2014.

[10]. Jyoti R Hallikhed, Mahesh R.K," VLSI Implementation Of Fast Addition Using Quaternary Signed Digit Number System", International Journal Of Ethics In Engineering & Management Education , Volume 2, Issue 5, May 2015.

[11]. M Naveen Krishna, T Ravisekhar," Fast Arithmetic Operations With QSD Using Verilog HDL ", International Journal Of Engineering Science And Innovative Technology (IJESIT) Volume 3, Issue 4, July 2014.

[12]. Bhavya Sree Kotte1 S Saleem Malik," Design Of Quaternary Signed Digit Adder", IJSRD - International Journal For Scientific Research & Development| Vol. 3, Issue 06, 2015.

[13]. G.Manasa1, M.Damodhar Rao2, K.Miranji," 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, Article 09455, October 2014.

[14]. Sachin Dubey1, Reena Rani2, Saroj Kumari3, Neelam Sharma," VLSI Implementation Of Fast Addition Using Quaternary Signed Digit Number System", IEEE International Conference On Emerging Trends In Computing, Communication And Nanotechnology,2013.

[15]. Aditya Sharma.," FPGA Implementation Of Quaternary Adder/Subtractor&Logic-Unit", International Journal Of Engineering Research & Technology (IJERT)\ ISSN: 2278-0181 Www.Ijert.Org Vol. 2 Issue 6, June - 2013.

[16]. Songpol Ongwattanakul, Phaisit Chewputtanagul, David J. Jackson, Kenneth G. Ricks," Quaternary Arithmetic Logic Unit On A Programmable Logic Device",2014.

[17]. R.Mohithreddy, DR.V.Ushashree, B. Kishore Kumar," An Efficient Implementation Of Signed Multiplier Using QSD Number System In VLSI", International Journalofreview In Electronics And Communication Engineering Volume 3, No 5 October 2015.

[18]. C.V.Sathish Kumar (1), P.Jaya Rami Reddy," Implementation Of A Fast Adder Using QSD For Signed And Unsigned Numbers ", International Journal Of Science, Engineering And Technology Research (IJSETR), Volume 3, Issue 11, November 2014.