High Speed 4 Bit QSD Addition / Subtraction

Shristi Arora
M.Tech Scholar , VLSI design , JECRC University, Jaipur , shristiarora90@gmail.com

Abstract :- The need for high speed digital circuits became more prominent as portable multimedia and communication applications incorporating information processing and computing. The drawback of modern computers lead to the deterioration in performance of arithmetic operations such as addition, subtraction, division, multiplication on the aspects of carry propagation time delay, high power consumption and large circuit complexity. This system explores the carry free n digits addition/subtraction as the carry propagation delay is most important factor regarding the speed of any digital system. In this paper , we are introducing the reversible logic gates based 4 Bit QSD Adder/Subtractor . For the fast operation of QSD, we apply pipeline also so that delay can be reduced.

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

1. INTRODUCTION

In various computers & other processors, adders are implemented in ALUs and also in other portions of processors for calculating the address, table indices & same kind of operations.

Even though adders can be generated for various numerical presentations like excess-3 or binary-coded decimal, most of the basic adders work over binary digits. In the cases were negative numbers are presented through two's or one's complement, it is necessary to transform an adder to adder-subtractor. Some other signed number presentations need a more complicated structurized adder.

1.1. Half Adder

The half adder adds up two single binary digits that are termed as A & B. it produces to outcomes, Carry (C) & Sum (S). This carry signal leads to an overflow in the subsequent digits of multi-digit addition. The value obtained from this sum is S + 2C. The most generalized design of half adder is displayed on the right hand side that is comprised of XOR gate for S & an AND gate for C. By adding an OR gate for combination of the carry outcomes, a full adder is made by combining two half adders.

The half adder adds two input bits and generates a carry and sum, which are the two outputs of a half adder. The input variables of a half adder are called the augends and addend bits. The output variables are the sum and carry.

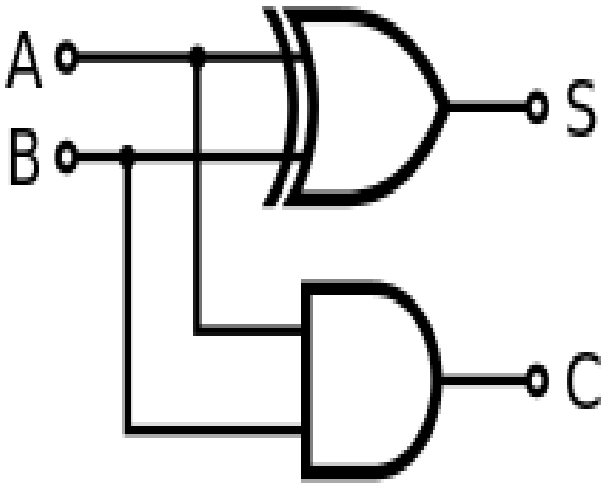

Figure 1:- Half Adder

INPUT		OUTPUT	
A	**B**	**C**	**S**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

Table 1:- Half adder truth table

1.2 Full Adder

A full adder is implemented for adding binary numbers & accounts where values are carried in & out as well. A one-bit full adder adds up three 1-bit numbers that are termed as A, B &C_{in}; where A & B are referred as operands and C_{in} is the bit that is carried in from past least significant level. The full adder is generally a component in cascade of adders that adds up 8, 16, 32 etc. bit binary digits. This circuitry generates a 2-bit output, sum & output carry which is presented by signals C_{out} & S, where

$$sum = 2 * C_{out} + S \quad (1)$$

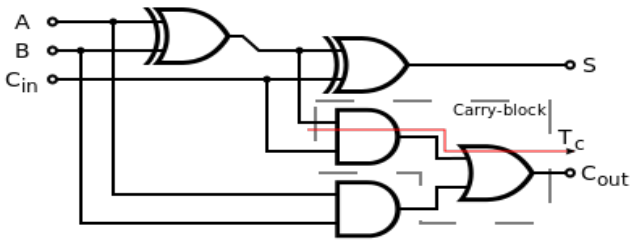

Figure 2:- Full adder

A full adder can be implemented in many different ways such as with a custom transistor-level circuit or composed of other gates. One example implementation is with

$$S = A \text{ xor } B \text{ xor } C_{in} \quad (2)$$

and

$$C_{out} = (A.B) + (Cin .(A \text{ xor } B)) \quad (3)$$

In such implementation, the final OR gate before carry-out outcome may get replaced by an XOR gate without making any transformations to resulted logic. By making use of only two forms of gates is easy if the circuitry is implemented by deployment of basic IC chips that are comprised of only a single gate type for each chip.

A full adder is formulated by combining two of the half adders by linking A & B to input of one half adder, that is linked to sum from that to an input to 2nd adder, linking C_i to other input & OR to the two carry outputs. The critical path in a full adder goes through both of the XOR gate & ends over the sum bit S. it is presumed that XOR gate needs 3 delays to get complete where delay is inflict by critical path of full adder which is equal to

$$T_{F.A} = 2 \, . \, T_{XOR} = 2.3D = 6D \qquad (4)$$

The subcomponents of carry block is comprised of 2 gates & so it has a delay of

$$T_C = 2D \qquad (5)$$

1.3 Quaternary Numeral System

Quaternary is taken as a base-4 number system. The numbers 0, 1, 2 & 3 are used to present any of the number. The number 4 is the maximum digit in subsidizing range and one of the number which is a high composite number & a square as well (as other is taken as 36), that makes quaternary as a rational selection as a base over this scale. Since it is double in the size than binary, still radix economy of both is same. Though, it doesn't work fine in localizing prime numbers (further best is the primordial base six, senary).

Various kind of characteristics like capability of presenting a real number having canonical presentation (that is almost unique) & presentation of rational & irrational numbers are been shared by the quaternary with all of the fixed radix numerical systems. Look over binary & decimal topic for explanation of these characteristics.

Arithematic operations have a vital role in several digitized systems like process controllers, computers, image processing & signal processors computer graphics. Latest improvisations in the techniques of integrated circuitries made the bigger circuits based over arithmetic functions to be implemented over VLSI [1]. Though such arithmetic functions are still dealing with issues like limited bits, time delay in propagation & complicacy in circuit. Now, flexibility in FPGAs has also supported enhancement in customized hardware giving high ratio of performance. By choosing arithmetic algorithms that suits FPGA technology & implementing optimal mapping techniques, high performance FPGA application ban be produced [7].

High speed QSD arithmetic and logical unit that has the ability of carry free addition, borrow free subtraction, multiplication functions & up-down count. The operation of addition and subtraction by QSD incorporates a defined number of inters for any of the size of an operand. The signed digit number system provides the chances of carry free addition. QSD Adder/ QSD Multiplier circuitries are designed in order to execute arithmetic functions at a high speed. In the number system of QSD, carry chain propagation is eliminated that further leads to minimization of computational time. It further improvises speed of machine [2].

1.4 Quaternary Signed Digit Numbers

QSD digits are present by making use of 3-bit 2's complement notation. Each digit is presented by:

$$D = \sum_{i}^{n} x_i 4^i \qquad (6)$$

Here x_i can be taken as any of the value from the set of {3, 2, 1, 0, 1, 2, 3} for generating a suitable decimal representation. A QSD negative number is obtained by complementing a QSD positive number which means the 3 = -3, 2 = -2, and 1 = -1. For digitized implementation, bigger sized digits like 64, 128 and more can be utilized trough a consistent delay. A signed digit number system constituted over higher radix value like QSD helps in more information storage density, low complicacy, less system constituents & low cascading gates & operations. This methodology helps in implementation of area effective & high speed adders & multipliers.

For Example

$$(1\bar{2}\bar{3}3)_{QSD} = (23)_{10}$$
$$= 1*4^3 + 2*4^2 + 3*4^1 + 3*4^0$$
$$= 64-32-12+3 = 23$$
$$= (\bar{1}23\bar{3})_{QSD} = (-23)_{10}$$

1.5 Adder Design

A faster Arithmetic set helps in elaboration of application domain for faster multipliers in processing of digitized signals, modular exponential, matrix inversion, calculation of Eigen values, digit filters etc. Square roots, calculation of reciprocals, inverse square toots & other elementary functions through implementation of small sized tables, small multipliers & for few functions, a final multiplication is obtained. Addition is taken as a cardinal operation in the field of digital computing. As the size of digits become large, a need of carry free addition is there. This carry free addition can be obtained by exploitation of redundancy in QSD numbers & QSD addition.

$$(6)_{10} = (12)_{QSD} = (2\bar{2})_{QSD} \qquad (7)$$

Two steps are involved in the carry free addition. In the first step, an intermediate carry & sun from augends & addend is produced. In the next step, intermediate sum of present digit & carry from lower significant digit is combined [3] [6]. There are two rules defined for prevention of carry from rippling. As per the first rule, value of magnitude of intermediate sum must be equal to or less than 2. As per the 2nd rule, value of magnitude of carry should be equal or less that 1. Further, magnitude of outcome obtained in the second step must not be higher than 3 that is presented by a single digit QSD number. Thus, no additional carry is needed. In the 1st step, the possible input pairs of augends & added are taken into account.

Sum	Intermediate Carry	Intermediate Sum
-6	-1	-2
-5	-1	-1
-4	1	0
-3	-1	1
-2	0	-2
-1	0	-1
0	0	0
1	0	1
2	0	2
3	1	-1
4	1	0
5	1	1
6	1	2

Table 2 :- The Intermediate Carry And Sum Between -6 To 6

Both of the inputs & outcomes can be encoded in 3-bit 2's complement binary number type. Mapping in the addend, augends, inputs & outputs, sum & intermediate carry are presented in binary form Table 2. As the value of intermediate carry remains in between -1 & 1, only 2-bot binary

presentation is needed. At last, five 6-variable Boolean expressions can be extracted.

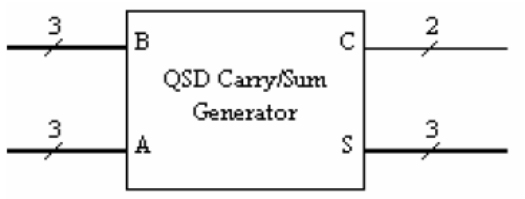

Figure 3. The intermediate carry and sum generator

In the 2nd step, sum generator & intermediate carry from lower significant digit is added to sum of current digit for generating the last final outcome no carry is produced by the addition in this step as carry-in coming from lower digit s absorbed by present digit. All the possible sets of summation generated in between sum & intermediate carry are presented.

Figure 4. The second step QSD added.

It is possible to extract three 5-variable Boolean expressions. N-QSD carry & sum generators along the n-1 second step adders is needed for implementing an n-digit QSD adder that is presented in the outcomes which comes out to be an n+1 digit number. The outcome obtained by this addition lie in between -3 to 3. As taking a carry is not permitted in this step, the outcome is a single digit QSD output. The inputs are, intermediate carry & sum that are 2-bit & 3-bit binary respectively. The outcome obtained is 3-bit binary representation of QSD number [4] [5].

It is possible to extract three 5-variable Boolean expressions. the implementation of the Boolean equations is done with the help of FPGA tools. The compilation & simulation of Modelism and Leonardo spectrum design is performed. There are two steps in the addition for design of adders. The 2nd step adder is presented in figure 2. N-QSD carry & sum generators along the n-1 second step adders is needed for implementing an n-digit QSD adder that is presented in figure 3. The outcome obtained is n+1 digit number.

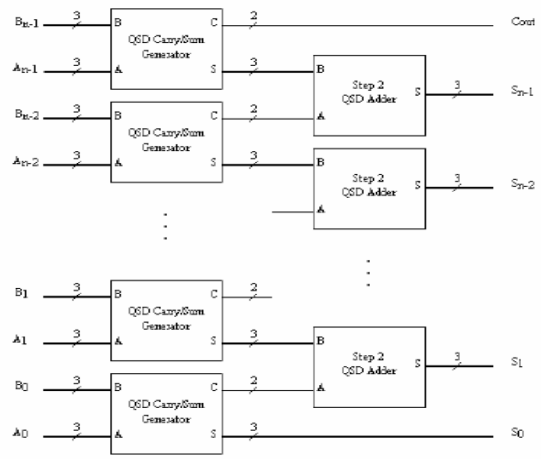

Figure 5 :- n-digit QSD adder

INPUT				OUTPUT				
QSD		BINARY		DECIMAL	QSD		BINARY	
A	B	A	B	SUM	S1	S0	S1	S0
-3	-3	101	101	-6	-1	-2	111	110
-3	-2	101	110	-5	-1	-1	111	111
-3	-1	101	111	-4	-1	0	111	000
-3	0	101	000	-3	-1	1	111	001
-3	1	101	001	-2	0	-2	000	110
-3	2	101	010	-1	0	-1	000	111
-3	3	101	011	0	0	0	000	000
-2	-3	110	101	-5	-1	-1	111	111
-2	-2	110	110	-4	-1	0	111	000
-2	-1	110	111	-3	-1	1	111	001
-2	0	110	000	-2	0	-2	000	110
-2	1	110	001	-1	0	-1	000	111
-2	2	110	010	0	0	0	000	000
-2	3	110	011	1	0	1	000	001
-1	-3	111	101	-4	-1	0	111	000
-1	-2	111	110	-3	-1	1	111	001
-1	-1	111	111	-2	0	-2	000	110
-1	0	111	000	-1	0	-1	000	111
-1	1	111	001	0	0	0	000	000
-1	2	111	010	1	0	1	000	001
-1	3	111	011	2	0	2	000	010
0	-3	000	101	-3	-1	1	111	001
0	-2	000	110	-2	0	-2	000	110
0	-1	000	111	-1	0	-1	000	111
0	0	000	000	0	0	0	000	000
0	1	000	001	1	0	1	000	001
0	2	000	010	2	0	2	000	010
0	3	000	011	3	1	-1	001	111
1	-3	001	101	-2	0	-2	000	110
1	-2	001	110	-1	0	-1	000	111
1	-1	001	111	0	0	0	000	000
1	0	001	000	1	0	1	000	001
1	1	001	001	2	0	2	000	010
1	2	001	010	3	1	-1	001	111
1	3	001	011	4	1	0	001	000
2	-3	010	101	-1	0	-1	000	111
2	-2	010	110	0	0	0	000	000
2	-1	010	111	1	0	1	000	001
2	0	010	000	2	0	2	000	010
2	1	010	001	3	1	-1	001	111
2	2	010	010	4	1	0	001	000
2	3	010	011	5	1	1	001	001
3	-3	011	101	0	0	0	000	000
3	-2	011	110	1	0	1	000	001
3	-1	011	111	2	0	2	000	010
3	0	011	000	3	1	-1	001	111
3	1	011	001	4	1	0	001	000
3	2	011	010	5	1	1	001	001
3	3	011	011	6	1	2	001	010

Table 3:- QSD Addition along with equivalent binary representations

1.6 QSD Subtraction

Subtraction process is similar to the addition process in QSD. In the QSD Subtraction, Carry(QSD Addition) is get replaced by QSD Borrow [8]. QSD numbers inputs and subtraction output table is shown in Table no 6.

In QSD subtraction, elimination of carry propagation chain gives the high speed of QSD subtraction in arithmetic operations. QSD numbers range has -3 to +3 , and the results of the QSD subtraction range is -6 to +6.

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	110
-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

Table 4:- QSD Subtraction with equivalent binary representation

Figure 4 is showing the Block diagram for the QSD addition and subtraction . $A_0,A_1,A_2.....A_{n-1}$ are the first input and $B_0,B_1,B_2.....B_{n-1}$ are the second input for the QSD addition or subtraction .

QSD Carry Sum generator produce the sum and carry for the Input QSD numbers A,B from the Table 3 for addition and Table 4 for Subtraction .

QSD Adder make the addition for QSD carry sum generator first block carry and second block sum . For perform the addition in QSD adder , using full adder equations .

$$Sum = A \text{ xor } B \text{ xor } C \quad .....(2)$$

$$Carry = (A \text{ and } B) \text{ or } (B \text{ and } C) \text{ or } (C \text{ and } A) \quad .....(3)$$

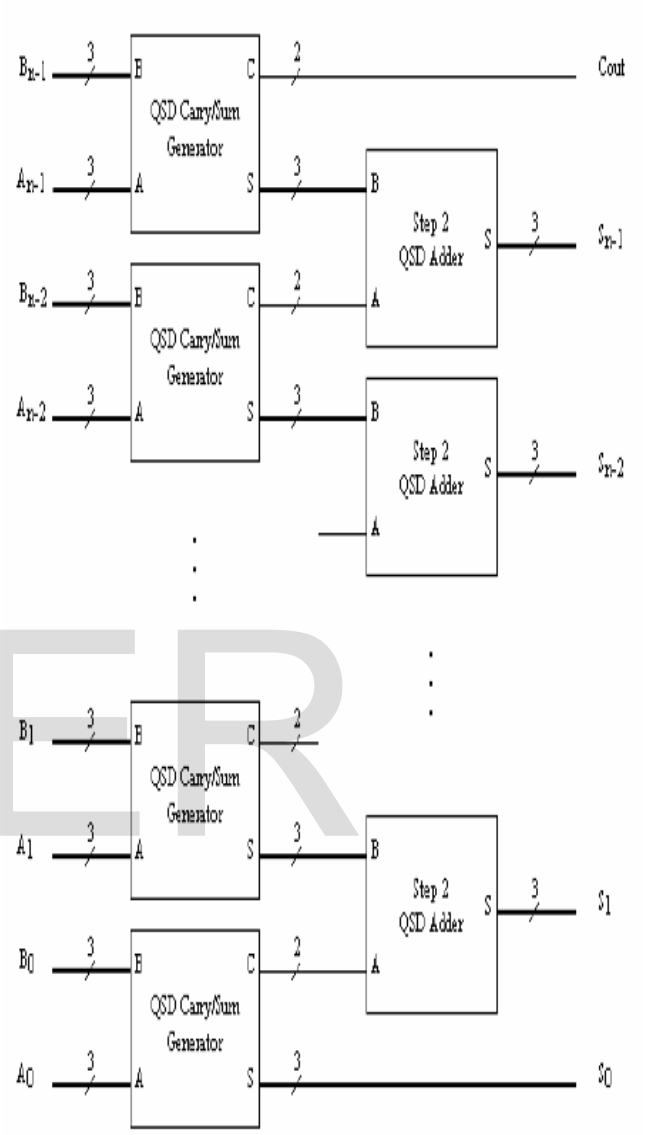

Figure 6 :- QSD Addition/ Subtraction Block Diagram

2. PROPOSED METHODLOGY

QSD works as an addition that is carry free and is implemented for enhancing the speed of system. Generally, QSD is implemented for carry & sum till now in a basic full adder for the purpose of addition.

2.1 Methodology 1

The Peres gate constituted over full adder is implemented for improving the performance under condition of delay. We are designing a full adder Peres gate based over the reversible logic.

The input vector is considered to be I (A, B, C) & the vector obtained for output is O (P, Q, R). The outcome is presented by P = A, Q = AB &

R = AB C. The cost of Quantum for Peres gate is 4. In the suggested design Peres gate is implemented as it has minimal quantum cost.

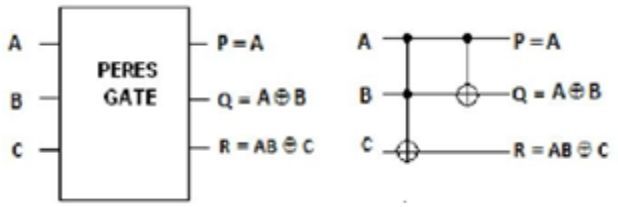

Fig 6 :- Peres Gate

$$P = A \quad \text{.....(4)}$$

$$Q = A \text{ xor } B \quad \text{.....(5)}$$

$$R = (A \text{ and } B) \text{ xor } C \quad \text{....(6)}$$

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 3 :- Peres gate Truth Table

A full-adder that applies two Peres gates as presented in the diagram. The quantum realization of this presents that cost of quantum is 8 where to Peres gates are applied. A 4*4 reversible gate termed as PFAG with quantum cost of 8 is applied to form the multiplier.

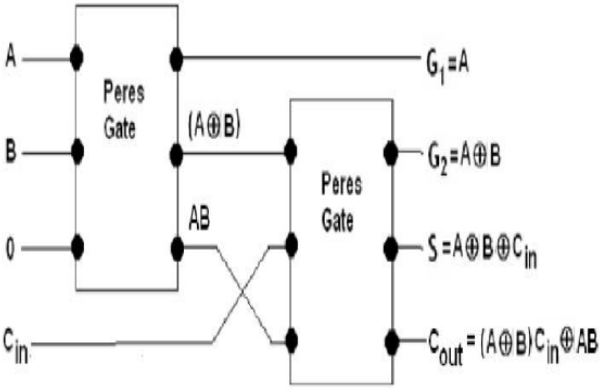

Figure 7 :- Full adder using two Peres gates

2.2. Methodlogy2

Here the method of pipelining is implemented for minimizing the delay in the circuitry. Initially, the pipelining is applied in QSD carry sum generator & the applied in bit transfer values for purpose of sum & carry. In this process we are able for transferring multiple clocks through only one clock.

3. RESULTS

In the results session , we will show the comparison of the 4 bit addition in between normal QSD addition and QSD adder by reversible logic gate . Table 5 is showing the comparison in between 4 bit normal QSD addition and Reversible Logic based QSD addition .

Design	Delay
QSD Adder (4 bit)	15.185ns
QSD Adder by Reversible Gate(4 Bit)	2.956ns

Table 5:- Delay comparison for 4 Bit QSD addition

Table 6 is showing the comparison of the 4 bit QSD subtraction and 4 bit Reversible logic based QSD subtraction .

Design	Delay
QSD Subtractor (4 Bit)	15.854ns
QSD Adder by Reversible Gate (4 Bit)	2.956ns

Table 6:- Delay comparison for 4 Bit QSD Subtraction

4. CONCLUSION

The propose QSD along the reversible logic gates has a better scope than the binary addresses for the purpose of low addition of delay. The operational speed will be increased by an efficient design for implementing it for purpose of multiplication or addition. The suggested fast speed adders constituted over quaternary signed digit number system along the reversible logic gates. In the QSD every digit is presented by a number from -3 to 3. The process of carry free addition & other functions works over large quantity of digits like 64, 128 or further more that are applied by a constant delay & low level of complicacy. The design of this circuitry is presented through FPGA tools. These designs are simulated by applying modelism software & are formulated by XILINX.

In VLSI we implement Reversible logic gates for improvising the performance for power of the circuitry. The delay, power & area get minimized after implementation of reversible logic gates in the QSD. We further implement Peres gates for minimizing the delay & area. The delay in a general QSD Adder will be 15.185ns & it becomes 2.956 ns after implementation of proposed methodology. The delay of 4 bit QSD Subtraction is 15.854 ns and delay for 4 bit QSD subtraction by reversible logic gate is 2.956 ns.

References

[1] Chao Cheng; Parhi and K.K. “High-Throughput VLSI Architecture for FFT Computation”, IEEE Transactions on Volume 54, Issue 10, Page 863 – 867, October 2007.

[2] I.M. Thoidis, D. Soudris, J.M. Fernandez , A. Thanailakis, “The circuit design of multiple-valued logic voltage-mode adders,” 2001 IEEE International Symposium on Circuits and Systems, pp 162-165,Vol. 4 , 2001.

[3] Hwang K. (1979) Computer Arithmetic Principles Architecture and Design. New York : Wiley.

[4] N. Takagi, H. Yasuura, and S. Yajima, “High Speed VLSI Multiplication Algorithm with a Redundant Binary Addition Tree, ” IEEE Trans. Comp., C-34, pp. 789-795, 1985.K. Elissa, “Title of paper if known,” unpublished.

[5] O. Ishizuka, A. Ohta, K. Tannno , Z. Tang, D. Handoko , “VLSI design of a quaternary multiplier with direct generation of partial products,” Proceedings of the 27th International Symposium on Multiple Valued Logic, pp. 169-174, 1997.

[6] J.U.Ahmed, A.A.S. Awwal, "Multiplier design using RBSD number system", Proceedings of the 1993 National Aerospace and Electronics Conference, pp. 180-184, Vol. 1, 1993.

[7] A. Avizienis, "Signed-Digit Number Representation for Fast Parallel Arithmetic, "IRE Transaction Electron. Comp., EC-10, pp. 389-400, 1961.

[8] M. E. Louie and M. D. Ercegovac," On Digit -Recurrence Division Implementations for Field Programmable gate Arrays" In Proc. Of the 11th symposium on Computer Arithmetic, PP. 202-209, Canada.