Realization of Carry-Free Adder Circuit Using FPGA

Shruti Jain

Abstract The circuit complexity and propagation delay are the main concern in the design of the digital circuit. In the binary system, as the number of bits increases, the computation speed is limited by the propagation of carry. Consequently, it offers low storage density, large complexity, and $O(n)$ carry producing delay in n-bit base 2 application problems. For less complexity and higher information storage density, higher radix number system can be used. Quaternary signed digit (QSD) number system performs carry-free addition and borrows free subtraction. The QSD number system requires a various set of prime modulo-based logic elements for each arithmetic operation. Arithmetic operations on a large number of bits (64, 128, 256, 512, or more) can be implemented with less complexity and constant delay that increases linearly. The data processing speed is limited in the base 2 number system because of the generation of carry peculiarly as the number of bits keeps on increasing. The QSD addition eliminates carry from addition and delayed addition. The authors have also worked on minimizing the delay at a particular frequency. For all these designing and calculations, Xilinx 14.7 software has been used. In this paper, authors have designed the ripple carry adder, carry-save adder, and carry-free adders and achieve 1.075 ns, 1.673 ns, and 1.878 ns delay, respectively.

Keywords QSD · Carry-free adder · Carry-save adder · RCA · Delay

1 Introduction

Operations involving addition, subtraction, and multiplication form the basic arithmetic operations which are extensively used and thus perform a significant part in many digital electronic devices such as signal mainframes and computers [1, 2]. The arithmetic units employing quaternary signed digit (QSD) number system have been a topic of great interest for many researcher scholars [3, 4]. High-performance arithmetic is very crucial as the adders used in the system decide the pace of the data

S. Jain (✉)
Jaypee University of Information Technology, Solan, Himachal Pradesh, India
e-mail: shruti.jain@juit.ac.in

A. K. Manocha et al. (eds.), *Computational Intelligence in Healthcare*, Health Information Science, https://doi.org/10.1007/978-3-030-68723-6_3

processing machine. It also handles a component for amalgamation rest of the calculation assignments [5, 6]. Digital signal processing is the main application of adder where it is used for the execution of FIR and IIR algorithms [7, 8]. Arithmetic operations still experience problems like the number of bits are limited, delay in producing time, and complication of the circuit. The time lag in an adder is established through the carry chain. Some of the drawbacks of BSD number systems are its computational speed that constraints generation and production of carry peculiarly when the number of bits is extended. Accordingly, it contributes to large complications and less cache denseness. Arithmetic which removes carry can be accomplished using a number system with higher radix such as (QSD) [9–11]. In QSD, there is a reduction of computational time due to the elimination of the carry propagation chain; this enhances the speed [12–14]. The QSD subtraction/addition operation utilizes a fixed number of operand size [15, 16]. Signed integer portrayal can be used to conclude the fast addition of integers because it can eradicate carry. The portrayal of a signed-integer QSD number is expressed by using Eq. (1):

$$D = \sum_{n-1}^{i=0} X_i 4^i \tag{1}$$

Authors in [17] reviewed VHDL for quaternary signed adder system. Authors in [18] designed and implemented QSD adders for arithmetic operation. In [19], implementation of large digits of digital like 64, 128, or large numbers can be implicated with the same amount of delay. Authors in [20] explained that with the increase of the radix, the redundancy usually increases in the quaternary (base 4) number system. In [21], multivalued logic is used, design of a paradigm of digital to analog converter circuit using an operational amplifier to substantiate the approach. An author in [22] designed the QSD cipher notation system addition using delayed addition technique.

Earlier, the other number systems were propagating chains, and thus the speed was limited. A lot of research was done on higher number systems, authors in this paper work on the quaternary signed digit. Considering digital logic in the field of VLSI, an optimal solution for the problem in which carry and borrow free addition and subtraction respectively of the quaternary signed digit can be calculated. Using QSD, propagation chains are eliminated and also prevent rippling of carry. QSD numbers are less complex and use 25% less space than binary signed number systems to store numbers. QSD number systems offer not only simple logic but also higher storage density. This paper proposes a tremendous productive QSD adder accomplished of addition and subtraction without the involvement of carry and borrows, respectively, which was simulated in Xilinx using Verilog HDL. In this paper, authors have also designed RCA, carry-save adder, and carry-free adder, and the results are compared with the existing state-of-the-art technique.

Section II of this paper explains the design of the QSD adder, and Section III explains the results of the proposed QSD adder and design of different adders, namely, RCA, carry-select adder, and carry-free adder followed by conclusion and future work.

2 Proposed Methodology

Arithmetic operations have a vital role in several digitized systems like process controllers, computers, image/signal processing, and computer graphics [23, 24]. The latest improvisations in the techniques of integrated circuitries made the bigger circuits based on arithmetic functions to be implemented over VLSI, although such arithmetic functions are still dealing with issues like limited bits, time delay in propagation, and complicacy in the circuit. The flexibility in FPGAs has also supported enhancement in customized hardware giving a high ratio of performance. By choosing arithmetic algorithms that suit FPGA technology and implementing optimal mapping techniques, a high-performance FPGA application can be produced. High-speed QSD arithmetic and a logical unit have the capability of doing various operations like carry-free addition, borrow-free subtraction, multiplication functions, and up-down count. The operation of addition and subtraction by QSD incorporates a defined number for any size of an operand [25, 26]. QSD adder is designed to execute arithmetic functions at a high speed as shown in Fig. 1. In the QSD number system, carry chain propagation is eliminated that further leads to the minimization of computational time. It further improvises the speed of the machine [27].

Quaternary is taken as a base-4 number system [4, 5]. The numbers 0, 1, 2, and 3 are used to present any of the numbers. The number 4 is the maximum digit in subsidizing range and one of the number which is a highly composite number and a square as well, 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 the same, although it doesn't work fine in localizing prime numbers. There are two different approaches to convert a decimal number to QSD: long division method and base 2 signed digit (2's complement form). One digit QSD number can be portrayed using a 3-bit base 2 proportionate shown in Table 1.

To novitiate n-bit base 2 data into its proportionate q-digit QSD information, this comprises of the n-bit base 2 information that should be translated into $3q$-bit base 2 information. For a change from base 2 to QSD, the required cardinal portrayal of base-2 bits is shown in Eq. (2):

$$n = 3q - \{1 \times (q-1)\} \tag{2}$$

To attain this, odd bit from LSB to MSB (3rd, 5th, 7th bit, etc.) is split into two different parts, but the MSB bit cannot be divided. Algorithm 1 shows the steps for finding the QSD number from a negative decimal number.

Algorithm 1: QSD number from a negative decimal number Input: Any decimal number

Output: QSD number

Step 1: Consider a negative number $(-109)_{10}$.
Step 2: Convert into base 2 signed bit representation $(1101101)_2$.
Step 3: Conversion in its QSD proportion by considering an odd bit.

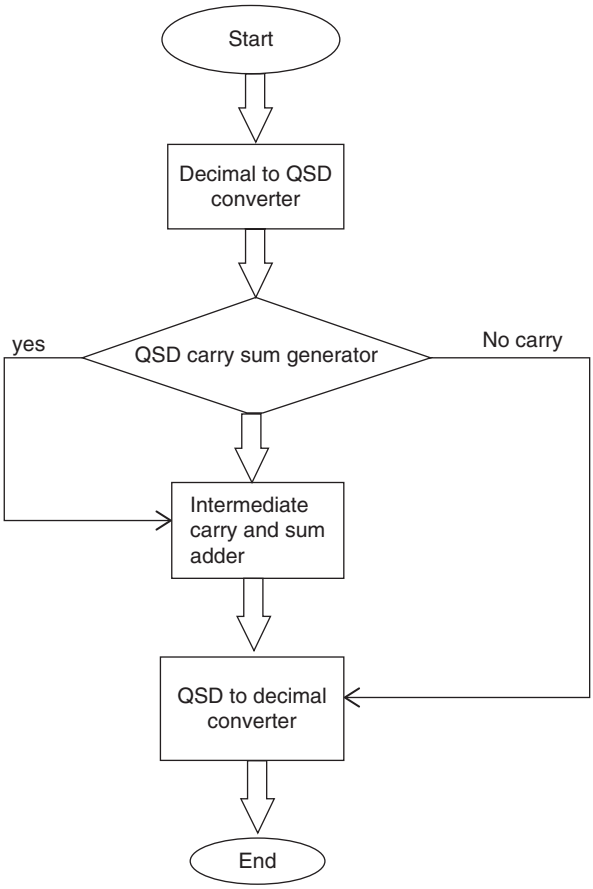
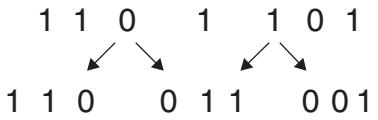


Fig. 1 Flow chart showing QSD addition

Table 1 Signed digit (2's complement form)

One digit QSD number	3-bit base 2 proportionate
$\bar{3}$	101
$\bar{2}$	110
$\bar{1}$	111
0	000
1	001
2	010
3	011

Step 4: If the odd bit is 0, then it is divided into 0 and 0, and if it is 1, then it is divided into 1 and 0 as illustrated:



Step 5: Conversion of binary into QSD by making a group of three. Final QSD number $(631)_4$.

More than one QSD representation can be used to represent the same decimal number. The number is chosen to avoid the fluctuation of the carry. The addition of the two QSD integers without carry can be done in the following two different tracks [5]:

1. In step 1, carry (intervening) and sum (intervening) are generated by the adder from the input digits.
2. In step 2, adder sums up the sum (intervening) of an ongoing integer with the carry (intervening) of a reduced indicative integer.

The fluctuation of carry can be removed by performing the following two rules in QSD addition [1]:

Rule 1: It states that the significance of the sum (intervening) must vary from −2 to +2.

Rule 2: It states that the significance of the carry (intervening) must vary from −1 to +1.

From the first step, the intervening sum and the intervening carry are obtained from the QSD adder whose province is from −6 to +6 as mentioned in the above rules. But using redundancy trait of QSD numbers, 655 is selected which satisfies the above-stated rules. When the intermediate sum whose range is from −2 to +2 as obtained in the second step of QSD adder is added with the lower significant digit intermediate carry whose province is from −1 to +1, the output of the addition cannot be larger than 3, i.e., it must lie in the province of −3 to +3; hence, there is no further requirement of carry as the addition output in this province can be depicted by an integer QSD [8–9].

Consider two inputs a_i (a_2, a_1, a_0) and b_i (b_2, b_1, b_0) considering the product yielding the intervening carry (*IC* as IC_2, IC_1, IC_0) and the intervening sum (*IS*: IS_2, IS_1, IS_0). The equations for the intervening sum (IS_2, IS_1, and IS_0) are expressed by Eqs. (3, 4, and 5), respectively:

$$IS_0 = a_0 \oplus b_0 \tag{3}$$

$$IS_1 = \left(a_0.b_0\right) \oplus a_1 \oplus b_1 \tag{4}$$

$$IS_2 = \left(IS_0.\left(a_1 \oplus b_1\right)\right) + \left(\overline{a_1}.\overline{b_0}.b_2\right) + \left(\left(a_0.b_0\right).\left(\overline{a_1}.\overline{b_1}\right).\left(a_2 + b_2\right)\right) + \left(a_0.b_0.a_1.b_1.a_2.b_2\right) \tag{5}$$

The equations for intervening carry (IC_2, IC_1, IC_0) are represented by Eqs. (6 and 7), respectively:

$$IC_2 = IC_1 = \left(a_2.b_2\right).\left(\overline{a_0.b_0.a_1.b_1}\right) + \left(\overline{a_1 + b_1}\right).\left(a_2.\overline{b_0} + \overline{a_0}.b_2\right) \tag{6}$$

$$IC_0 = IC_2 + \left(\overline{a_2.b_2}\right).\left(a_1.b_1 + b_1.b_0 + a_1.b_0 + a_0.b_1 + a_1.a_0\right) \tag{7}$$

The final sum (S_0, S_1, and S_2) is calculated by using Eqs. (8, 9, and 10), respectively:

$$S_0 = IC_0.\overline{IS_0} + \overline{IC_0}.IS_0 \tag{8}$$

$$S_1 = IC_1 \oplus IS_1 \oplus \left(IC_0.IS_0\right) \tag{9}$$

$$S_2 = IC_2 \oplus IS_2 \oplus \left(\left(IC_1.IS_1\right) + \left(\left(IC_1 + IS_1\right).\left(IC_0.IS_0\right)\right)\right) \tag{10}$$

The QSD integer ranges from −3 to +3, and when the two QSD integers are added, the result varies from −6 to +6. The product of all the feasible combinations of two numbers is illustrated in Table 2. If the range is exceeded by the decimal number, then there is a requirement of one more QSD digit. In the addition of two-digit QSD outcome, LSB is represented by the sum bit, and MSB bit is represented by carry.

Table 2 Intermediate sum and carry from −6 to +6

Sum	Possible QSD representations	QSD number	
		IC	IS
−6	$\overline{1}\,\overline{2}, \overline{2}\,2$	$\overline{1}$	$\overline{2}$
−5	$\overline{1}\,\overline{1}, \overline{2}\,3$	$\overline{1}$	$\overline{1}$
−4	$\overline{1}, 0$	$\overline{1}$	0
−3	$\overline{1}\,1, 0\,\overline{3}$	$\overline{1}$	1
−2	$\overline{1}, 2, 0, \overline{2}$	0	$\overline{2}$
−1	$0\,\overline{1}, \overline{1}\,3$	0	$\overline{1}$
0	0 0	0	0
1	$0\,\overline{1}, \overline{1}\,3$	0	1
2	$1\overline{2}, 0\,2$	0	2
3	$0\,3, 1\,\overline{1}$	1	$\overline{1}$
4	1 0	1	0
5	$1\,1, 2\,\overline{3}$	1	1
6	$1\,2, 2\,\overline{2}$	1	2

By scaling the two digits into intervening sum and intervening carry in such a way that the *n*th intervening sum and the $(n\text{-}1)^{th}$ intervening carry will not at all generate any propagating carry pair so that the rippling of the carry can be eliminated. Confining the representation of the QSD number according to the rules as defined above, the endmost addition will have no carry in the end.

3 Results and Discussion

In this paper, the QSD adder is designed and simulated in Verilog HDL of Xilinx Software 14.7. The circuit is implemented on the FPGA using the VIVADO tool. An FPGA is IC, a semiconductor device which can be reprogrammed by the customer according to his requirement after manufacturing. It comprises of an array of configurable logic blocks (CLBs) and switches (which form a connection between CLBs) [28, 29]. FPGA gives high-speed clock, good performance, and high bandwidth and facilitates simultaneously multiple operations. The FPGA is advantageous for implementation in real-time processing systems. The FPGA can remodel the digital design straight to ASICs. An FPGA consisting interconnections, I/O blocks, and reconfigurable logic, that differs from DSP and microcontroller processors. It is a better implementation solution in the latest digital systems due to its low NRE cost, radiation effects, power consumption, reconfigurable architecture, better performance, and ease of design. The design and architecture of the programmable routing circuit is the main issue and demand in FPGAs. Before the use of FPGA in portable electronic devices, various issues like static power consumption should be addressed. In recent years, FPGA is widely used in medical-related applications because of its flexible programming, low power consumption, easy transfer, and short development cycle.

Using the QSD adder, carry-free QSD adder, ripple carry adder (RCA), and carry-save adder are designed. The circuit is designed as shown in Fig. 2.

Results for ripple carry adder: RCA is considered the basic approach amid all the addition algorithms. An *N* bit RCA requires *N* number of full adders in series cascading. RCA and ripple carry subtractor are designed using the proposed QSD adder. The simulation results of the RCA and ripple carry subtractor are illustrated in Figs. 3 and 4, respectively.

In figures, a_0, a_1, a_2, b_0, b_1, b_2, and c_{in} are the inputs; s_3, s_2, and s_1 are the final turnouts; and c_{out1}, c_{out2}, and c_{out3} are the trajectory final turnouts. This result has been obtained by changing the C_{in} value from 0 to 1 in the test bench of the Xilinx code. The synthesis report shows the total number of LUTs and slices used during the simulation of the RCA circuit as tabulated in Table 3. The three different time lags total delay, the logic delay, and the route delay of the circuit are calculated. Table 3 shows the comparison of ripple carry adder with other existing work in terms of different parameters like LUT, slices, gates, and delay.

It has been observed that the design of the ripple carry adder by Prabha et al [16] shows 44.366ns delay. But the calculated delay of our proposed circuit is 1.075ns.

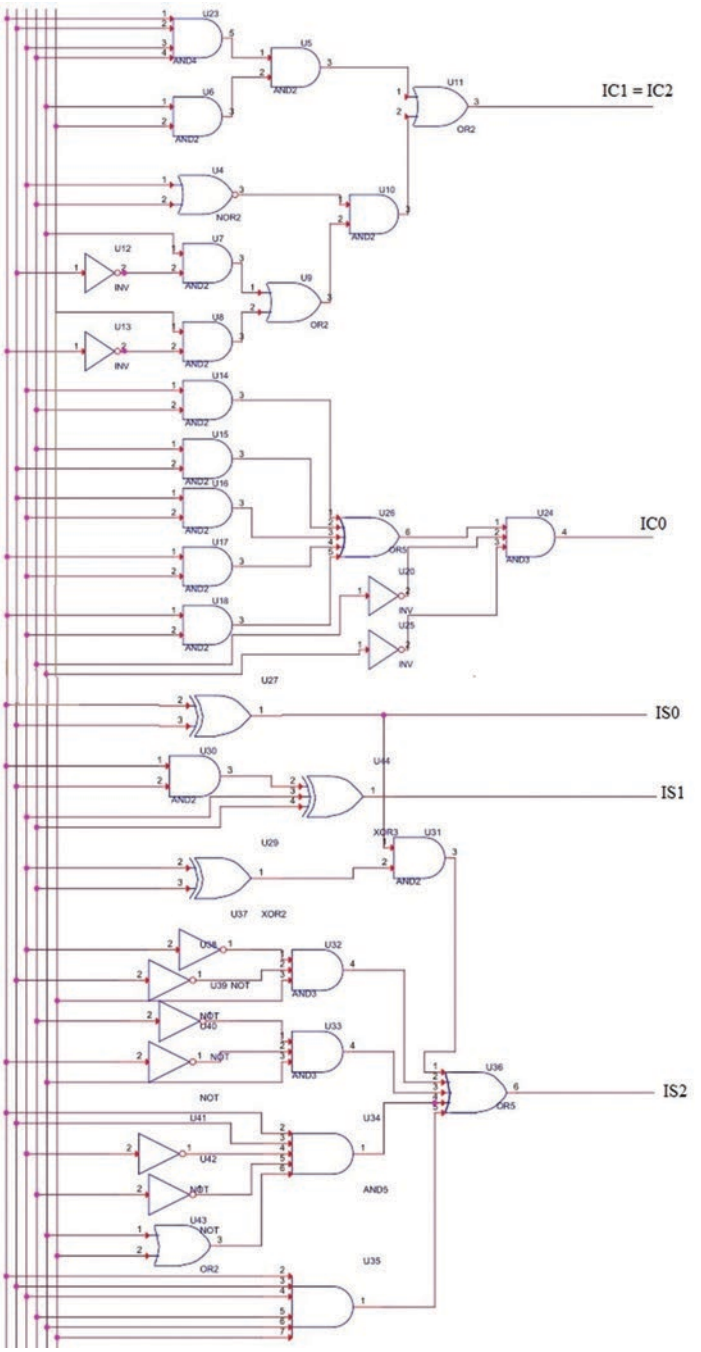


Fig. 2 QSD adder circuit

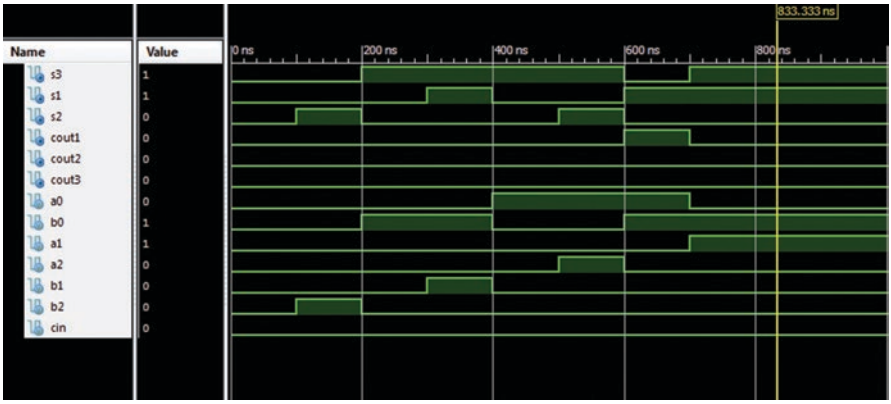


Fig. 3 Simulation result of RCA

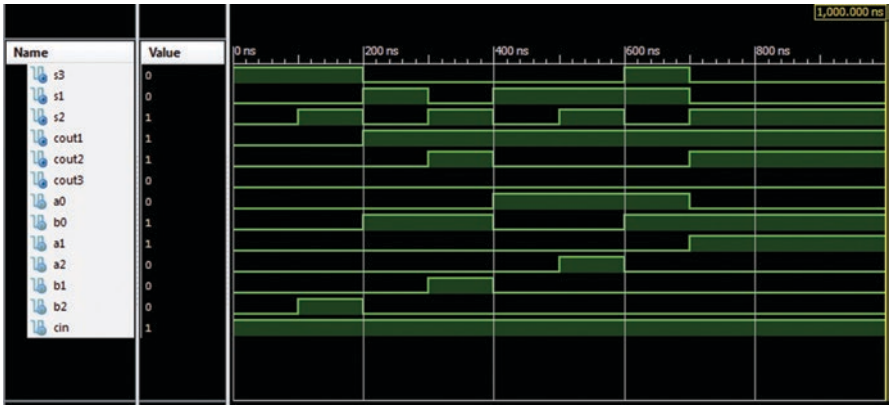


Fig. 4 Simulation result of ripple carry subtractor

Results for Carry-Save Adder: Carry-save adder is used for the fast calculation of addition. For getting sum bit and carry bit, bitwise XOR and bitwise AND operations, respectively, are performed. Finally, add them by shifting carry bit left by one place to sum bit up to produce the final answer. Carry-save adder is designed using the proposed QSD adder. The simulation results of the carry-save adder is shown in Fig. 5.

In figure, a_0, a_1, a_2, b_0, b_1, b_2, and c_{in} are the inputs; s_3, s_2, and s_1 are the final turnouts; and c_{out1}, c_{out2}, and c_{out3} are the trajectory final turnouts. This result has been obtained by changing the C_{in} value from 0 to 1 in the test bench of the Xilinx code. The synthesis report shows the total number of LUTs and slices used during the simulation of the CSA as tabulated in Table 4. Table 4 explains the different values obtained from the synthesis report which have been compared with various other results obtained after studying various research papers.

Table 3 Comparison table for ripple carry adder

Authors	LUT	Slices	Gates	Delay (ns)	Logic delay (ns)	Route delay (ns)
Prabha et al. [16]	11	4	15	44.366	11.28	45.393
Our work	5	5	15	1.075	0.98	0.977

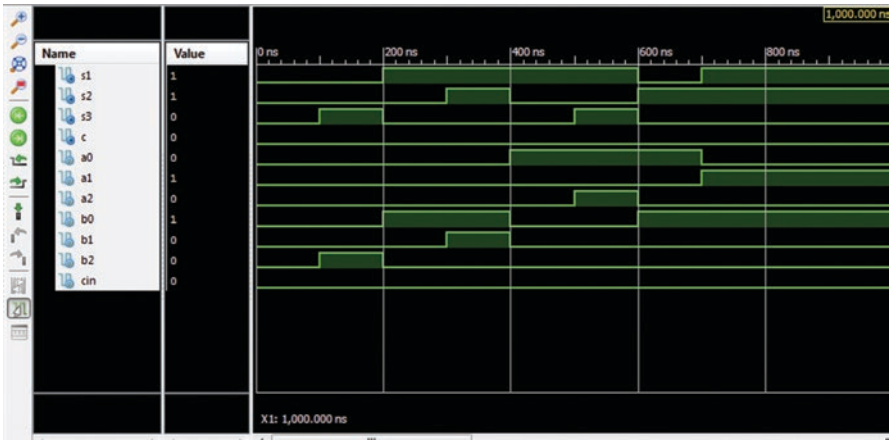


Fig. 5 Simulation result of carry-save adder.

Table 4 Comparison table for carry-save adder

Authors	LUT	Slices	Gates	Delay (ns)	Logic delay (ns)	Route delay (ns)
Kumar and Punniakodi 2013 [30]	13	9	20	1.433	0.289	1.144
Our work	5	5	20	1.673	0.195	1.478

It has been observed that the design of the carry-save adder by Kumar and Punniakodi (2013) [30] shows 1.433 ns delay. But the calculated delay in our proposed diagram is 1.673 ns.

Results Using Carry-Free Adder: The carry-free adder is designed and simulated. The simulation result is illustrated in Fig. 6. The performance parameters were calculated which are tabulated in Table 5.

In Table 5, different values obtained from the synthesis report have been compared with various other results obtained after studying various research papers. It has been observed that the design of the carry-save adder by Hussain and Rao (2014) [31] shows a 2 ns delay. But the calculated delay in our proposed diagram is 1.878 ns resulting in a 6.1% improvement than the other state-of-the-art technique.

This data is processed by FPGA, and the program is burned as shown in Fig. 7. Finally, the carry-free adder output is sent from FPGA to the PC for display.

Carry-free adder, RCA, and carry-save adders are designed, and the results are compared with other state-of-the-art techniques as tabulated in Table 6.

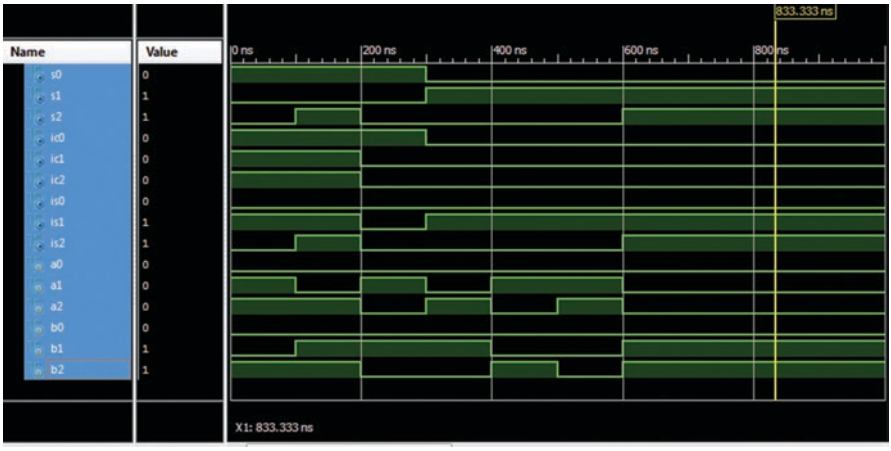


Fig. 6 Simulation result of carry-free adder (QSD)

Table 5 Comparison table for carry-free adder

Authors	LUT	Slices	Gates	Delay (ns)	Logic delay (ns)	Route delay (ns)
Hussain and Rao 2014 [31]	28	47	38	2	0.268	2.257
Our work	9	7	38	1.878	0.195	1.638

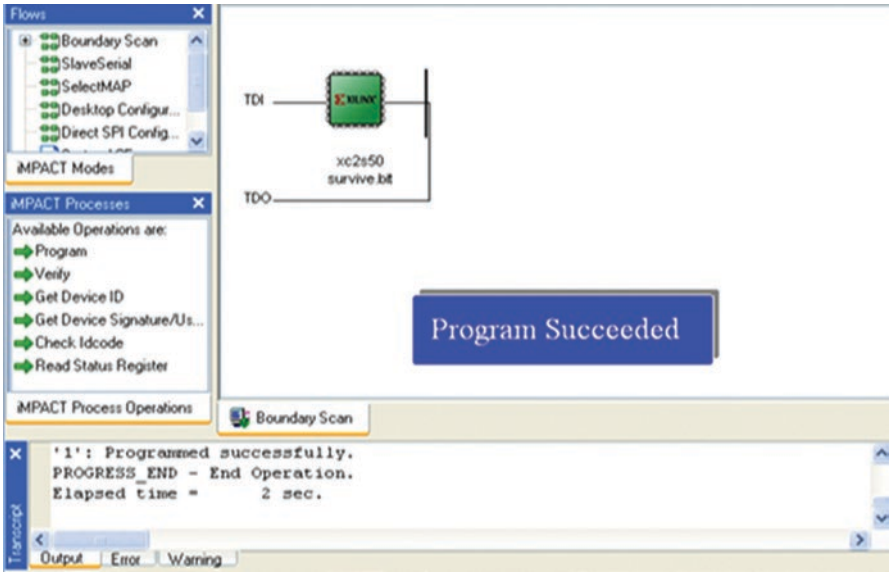


Fig. 7 Implementation of the proposed algorithm on FPGA

Table 6 Delay (ns) calculation

Authors	RCA	Carry-save adders	Carry-free adder
Prabha et al. [16]	44.366	–	10.113
Mitra et al. [32]	3.71	3.09	–
Proposed work	**1.075**	**1.673**	**1.878**

It has been observed that our circuit's results in 71% improvement in RCA circuit and 45.85% improvement in carry-save adder circuit with Mitra et al.

4 Conclusion

In this paper, authors have compared carry-free adder, RCA, and carry-save adders in terms of different performance parameters like delay, LUTs, slices, and gates. The suggested design of QSD generates a time delay of 1.878 ns. The proposed circuit consumes less energy and reduces time lag leading to better performance. It has been observed that our circuit's results in 71% improvement in RCA circuit and 45.85% improvement in carry-save adder circuit with other state-of-the-art techniques. As a result of which, this structure is applicable to be adapted for the implementation of a highly fulfilled multiprocessor which may consist of many processing components. This work can be counterfeited in the transistor level to minimize the delay and optimize the power by considering numerous low-power techniques.

References

1. K. Hwang, Computer Arithmatic Principles Architecture and Design. New York: Wiley.
2. Koren, Computer Arithmetic, New Jersey, Englewood Cliffs: Prentice Hall.
3. S. Dubey, R. Rani, S. Kumari and N. Sharma, "VLSI implementation of fast addition using quaternary Signed Digit number system," *2013 IEEE International Conference ON Emerging Trends in Computing,* Communication and Nanotechnology (ICECCN), Tirunelveli, 2013, pp. 654–659, https://doi.org/10.1109/ICE-CCN.2013.6528581.
4. 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.
5. A. N. Nagamani and S. Nishchai, "Quaternary High Performance Arithmetic Logic Unit Design," 2011 14th Euromicro Conference on Digital System Design, Oulu, 2011, pp. 148–153, doi: https://doi.org/10.1109/DSD.2011.23.
6. S Jain, P.K. Naik, R Sharma, "A Computational Modeling of cell survival/ death using VHDL and MATLAB Simulator", Digest Journal of Nanomaterials and Biostructures (DJNB), 4 (4): 863–879, 2009.
7. N Prashar, M. Sood, S. Jain, "Design and Performance Analysis of Cascade Digital Filter for ECG Signal Processing", International Journal of Innovative Technology and Exploring Engineering (IJITEE), 8(8), 2659–2665, 2019.
8. Kirti, H. Sohal, S. Jain, "FPGA implementation of Power-Efficient ECG pre-processing block", International Journal of Recent Technology and Engineering, 8(1), 2899–2904, 2019.

9. R. Rani, U. Agrawal, N. Sharma and L.K Singh, "High Speed Arithmetic Logical Unit using Quaternary Signed Digit Number System", *International Journal Of Electronic Engineering Research*, vol. 2, pp. 383–391, Number 2010, ISSN 0975-6450.
10. P. K. Dakhole and D.G. Wakde, "Multi Digit Quaternary adder on Programmable Device: Design and verification", *International Conference on Electronic Design*, pp. 1–4, 2008, 1–3 Dec.
11. M. W. Allam and M. I. Elmasry, "Low power implementation of fast addition algorithms," *Conference Proceedings. IEEE Canadian Conference on Electrical and Computer Engineering (Cat. No.98TH8341),* Waterloo, Ontario, Canada, 1998, pp. 645–647 vol.2, doi: https://doi.org/10.1109/CCECE.1998.685579.
12. R Thakur, S Jain, M. Sood "FPGA Implementation of Unsigned Multiplier Circuit Based on Quaternary Signed Digit Number System" , 4th IEEE International Conference on signal processing and control (ISPCC 2017), Jaypee University of Information technology, Waknaghat, Solan, H.P, India, pp 637–641, September 21–23, 2017.
13. W. Clark, J. Lian, "On arithmetic weight for general radix representation of integers". IEEE Trans. Inform. Theory, Vol. 19, pp. 823–826, 1973.
14. H. Shirahama and T. Hanyu, "Design of High Performance Quaternary Adders Based on Output-Generator Sharing", Proceedings of the 38th International Symposium on Multiple Valued Logic, pp. 8–13. 2008.
15. S.A.Dakhane, A.M.Shah, "FPGA Implementation of Fast Arithmetic UnitBased on QSD", International Journal of Computer Science and Information Technologies, Vol. 5(3), pp: 3331–3334, 2014.
16. A.Divya Prabha, V.Murali Dharan, M.Varatharaj, "Accomplishment of Formidable Agility QSD Adder for VLSI Application", IJRE, Volume 3, Issue 3, pp. 11–13, 2016.
17. P. S. Kamble, Suman M. Choudhary, "Review of VHDL Implementation of Quaternary Signed Adder System" 2012.
18. R. K. Kothuru, P. V. Ramana, "Design and Implementation of QSD Adders for Arithmetic Operation", International Journal of Professional Engineering Studies, Volume II/Issue 3/ June2014.
19. S. Kaur, S. Singh, "Design of Hybrid Quaternary Signed Digit (QSD) based divider Using VHDL", International Journal of Engineering Sciences & Research Technology,4(8), August, 2015.
20. K.Kuntal, S.Kumari, R.Rani, N.Sharma, "Implementation of QSD Adder using VLSI", Proceedings of IRF International Conference, 5-6 February 2014, Pune India. ISBN: 978-93-82702-56-6.
21. T Chattopadhyay and T Sarkar, "Logical Design of Quaternary Signed Digit Conversion Circuit and its Effectuation using Operational Amplifier", Bonfring International Journal of Power Systems and Integrated Circuits, Vol. 2, No. 3, December 2012 .
22. S.Mallesh, C.V.Narasimhulu, "Design of QSD Cipher notation System Addition using Delayed Addition Technique", International Journal of Ethics in Engineering & Management Education, Volume 1, Issue 10, pp: 1–4, October 2014.
23. J. Millman, C. Halkias, Integrated Electronics: Analog and digital circuits and systems. New York: McGraw-Hill, Inc.
24. V Patel, K.S. Gurumurthy, "Arithmetic operations in multivalued logic." International Journal of VLSI Design & Communication Systems (VLSICS), Vol. 1, No. 1, March 2010.
25. A. Leela Bhardwaj Reddy, V. Narayana Reddy, "VLSI Implementation of Fast Addition Subtraction and Multiplication (Unsigned) Using Quaternary Signed Digit Cipher notation System", 2(12), pp: 2061–2074, December 2015.
26. P.K. Dakhole, D.G. Wakde, "Multi Digit Quaternary adder on Programmable Device: Design and verification", International Conference on Electronic Design, pp. 1–4, Dec 2008.
27. T. Chattopadhyay, J.N. Roy, "Easy conversion technique of binary to quaternary signed digit and vice versa." Physics Express, Vol. 1, No. 3, pp. 165–174, 2011.

28. Sonal Pokharkar, Amit Kulkarni, **"**FPGA Based Design and Implementation of ECG Feature Extraction", International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE) Volume 1, Issue 12, May 2015.
29. A. Armato et al., "An FPGA Based Arrhythmia Recognition System for Wearable Applications," 2009 Ninth International Conference on Intelligent Systems Design and Applications, Pisa, 2009, pp: 660–664. https://doi.org/10.1109/ISDA.2009.246.
30. M.S. Kumar, S. Punniakodi, "Design and Performance Analysis of Various Adders using Verilog", International Journal of Computer Science and Mobile Computing Vol.2, Issue. 9, pp: 128–138, September 2013.
31. S.J.Hussain, K. S. Rao, "Design and Accomplishment of Fast Aggregation Using QSD for Signed and Unsigned Cardinal depictions", International Journal of Engineering Research, Volume No.3 Issue No: Special 2, pp: 52–54, 22 March 2014.
32. A.Mitra, A.Bakshi, B.Sharma, N.Didwania, "Design of a High Speed Adder", International Journal of Scientific & Engineering Research, 6(4), pp: 918–921, April-2015.