Design and Implementation of Radix 4 Based Arithmetic Operations

S. Saste[1] and A. Sawant[2]

[1]Dept. of Electronics & Telecommunication, Trinity College of Engineering & Research, Pune, India.
[2] Assistant Professor, Dept. of Electronics & Telecommunication, Trinity College of Engineering & Research, Pune, India.
{supriya.saste1992@gmail.com, anilsawant.22@gmail.com}

Abstract. Speed of the arithmetic operations often depends upon formation and propagation of carry. With the binary number system, the speed of arithmetic operations such as addition and subtraction is limited by formation and propagation of carry especially as the number of bits increases. The recent rapid increase in scale of integration is resulting in implementation of many sophisticated signal processing as well as video processing systems on VLSI chip in which multiplication is dominant operation. The performance of these systems is based on computation capacity and power consumption. This paper presents novel approach of multiplication scheme based on Radix 4 and implementation of addition/subtraction by using high radix system such as quaternary signed digit number system (radix-4).
Using a high radix system such as quaternary signed digit number, carry free arithmetic operations can be performed. The special set of prime modulo based logic for arithmetic operations is being used in such number system. In this paper, this high radix number system is used for faster implementation of addition and subtraction. This system has been designed and simulated using Xilinx 13.4 for 8x8 bit numbers.

Keywords: ***Booth's Algorithm, Radix 4, VLSI, QSD, Xilinx 13.4.***

1 Introduction

Arithmetic operations are widely used and play an important role in various digital applications [1] and hence the high speed arithmetic are essential as the speed of digital processor depends heavily on the speed of adders [2] as well as multipliers used in digital system. Limited number of bits, circuit complexity and delay are the problems, arithmetic operations suffer from [2]. In any digital system the arithmetic blocks serves as the basic building blocks for synthesis of all other arithmetic operations. Adder is one of the most commonly used arithmetic blocks in various electronic systems. Adders are used to perform algorithms like FIR, IIR [2][3][4] and act as the basic element of multiplication operation. Various types of adders are there viz., carry look ahead adder, carry save adder having their own limitations. The propagation delay associated with carry look ahead adder is less, but it is limited to small number of bits [2]. This paper presents implementation of three basic arithmetic operations viz., addition/subtraction and multiplication based on radix-4 scheme. Multiplication operation is implemented by using modified Booth's (Radix-4) algorithm and higher radix system i.e. quaternary signed digit (QSD) number system (Radix=4) is used for implementation of addition/subtraction operations.
Many times addition/subtraction operations carried out by binary number system have speed limitation due to formation and propagation of carry. Quaternary signed digit (QSD) adders are capable of performing carry free addition and borrow free subtraction [4][18][19] by using QSD number system. In QSD number system the carry propagation chain is eliminated which in turn reduces computation time [5]. Quaternary Signed Digit have major contribution in higher radix (=4) carry free arithmetical operation [6]. The addition/subtraction operations based on this number system employs a fixed number of min-terms for any operand size [5-7][22].

B. Iyer, S. Nalbalwar and R. Pawade (Eds.)
ICCASP/ICMMD-2016. Advances in Intelligent Systems Research.
Vol.137, Pp. 800-809.

Multiplication is another most important arithmetic operation which is used in high performance systems such as microprocessors, digital signal processors and multimedia applications [8][9][12]. Previously multiplication was done by repetitive sequence of other two basic arithmetic operations viz., addition, and subtraction along with shift operations. Hence, multiplication is repetitive addition of numbers. The „multiplicand" is a number which is to be added and number of times it is added is called as "multiplier". The repetitive addition method to employ multiplication is comparatively slow.

Multiplication is mainly performed in three different stages: In first stage, partial products are generated. The next stage i.e. stage two deals with reduction of partial products and finally in stage three all the partial products are summed together to get the final result of multiplication operation. The fundamental principle of multiplication is generation of partial products and accumulation of partial products [9]. Multiplication can be performed both on signed as well as unsigned numbers [10]. However signed multiplication is careful operation. Signed numbers cannot be multiplied in same manner as that of the unsigned numbers. Here the Booth's algorithm comes in. The motivation of Booth's multiplication scheme is to increase the speed of multiplication process. As compared to conventional methods modified Booth's (Radix-4) multiplication helps to reduce the number of iteration steps and results in faster computation. In this paper we present 8 bit multiplication by using modified Booth's (Radix-4) algorithm and its implementation on hardware platform.

2 Modified Booth's (Radix-4) Algorithm

The main version of Booth's algorithm (Radix-2) had two drawbacks:

1) With the invariability of add/subtract operations, the algorithm became inconvenient while designing parallel multipliers.

2) If there is a string of isolated 1s, the algorithm becomes inefficient [17].

The drawbacks enlisted in Booth's algorithm are overcome by using Modified Booth's Algorithm (Radix-4).

The number of bits multiplier/multiplicand is composed of gives exact number of partial products generated in multiplication operation. So, to perform the addition of partial product is main bottleneck in multiplication operation and considered as the important factor to speed up multiplication. In Booth's re-coding (Radix-2) algorithm, if 2 "n" bit numbers are multiplied then "n" partial products will be generated. The desired high speed can be achieved if the partial products are reduced. Modified Booth's (Radix-4) Algorithm uses the technique of partial product reduction to speed up multiplication operation. So, if 2 even "n" bit numbers are multiplied, the number of partial products generated is "n/2" and if "n" is odd, number of partial products are "n+1/2". Thus, in Radix-4 the number of partial products is reduced to half. To have high speed multipliers, Modified Booth's Algorithm is an ultimate solution. This algorithm scans strings of three bits at a time.

The numbers of steps involved in Radix-4 multiplication algorithm are shown below:

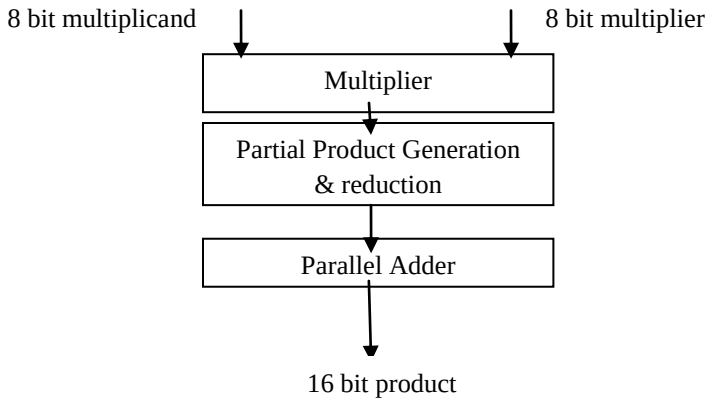


Fig.1 Radix 4 multiplication steps

In Modified Booth's (Radix-4) Algorithm, the multiplicand is re-coded based on bits of multiplier which can take any value from ± 1, ± 2 or 0. Three bits of multiplier are compared at a time, by using overlapping technique. Similar to Radix-2, we have to group bits of multiplier starting from LSB for which first block only uses two bits, considering third bit as zero. Following steps are to be performed in order to generate re-coded multiplier of Radix-4:

a) In order to ensure that n is even, extend the sign bit 1 position (if necessary).
b) Add a 0 to right of the LSB of multiplier.
c) Based on the value of each re-coded bit, each partial product will be 0, +M, -M, +2M or 2M.For Radix-4 bit pairing is done as shown below:

$$\overline{0\ 11}\ 1\ \overline{0\ 0\ 1}\ \overline{1\ 0}$$

Following table depicts the functional operation of Radix-4 Booth encoder:

Table I. Radix-4 Encoding Rules

Xn	Xn+1	Xn-1	Recoded Bits	Operations Performed
0	0	0	0	0
0	0	1	+1	+1M
0	1	0	+1	+1M
0	1	1	+2	+2M
1	0	0	-2	-2M
1	0	1	-1	-1M
1	1	0	-1	-1M
1	1	1	0	0

In above table "M" is nothing but multiplicand

Consider multiplicand and multiplier is composed of 4 bits respectively.
Multiplicand 1100
Multiplier 1010

So, according to Radix 4 recoding rules, partial products obtained are:

PP0= 1000

PP1= 0100

From above example, it can be concluded that if there is 4 bit number, obtained partial products are 2 i.e. for "n" bit number we get "n/2" partial products. Since, the number of partial products are reduced, speed of multiplication process increases. Final product of multiplication is obtained by adding partial products.

3 Quaternary signed digit number system

Quaternary signed digit number system is higher radix system for which radix used is 4. The degree of redundancy usually increases with increase of radix. The numbers in this system are signed digit numbers having digit set as: {-

3,-2,-1, 0, 1, 2, 3}. For every signed digit decimal number D can be represented in terms of an "n" digit quaternary signed digit number represented as,

$$= \sum_{=0}^{-1} 4 \qquad \ldots 1$$

y_i can be any value from QSD digit set i.e. from -3 to +3. For 1 digit QSD representation we need 3-bit binary number. The QSD numbers and its equivalent binary numbers are listed below:
-3=101, -2=110, -1=111, 0=000, 1=001, 2=010 3=011

3.1 General Block Diagram and Technique for Conversion of Binary to QSD

In QSD number system carry propagation chains 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, thus addition result of two QSD numbers varies from -6 to +6. For performing any operation in QSD, first convert the binary number into quaternary signed digit. The basic idea of QSD (Radix=4) addition is depicted in following figure:

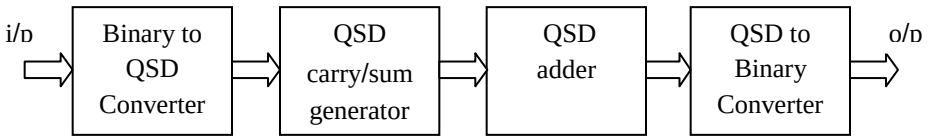
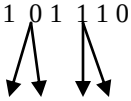


Fig.2 General Block Diagram of QSD Addition

1 digit QSD number can be represented by using a 3-bitbinary equivalent. In order to convert any binary number into its equivalent QSD number we must split the 3rd,5th,7 th bit i.e. odd bit(from LSB to MSB)into two portion .We cannot split the LSB & MSB bit this is major thing. If the odd bit is 0 then it is splited into 0&0 and if it is 1 then it is splited into 1&0.for example:$(1101110)_2$.

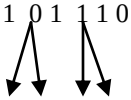
1 1 0 1 1 1 0



1 1 0 0 1 1 0 1 0

After splitting the odd bits of binary number, groups of three bits are formed starting from LSB and their equivalent QSD numbers are assigned. The equivalent QSD number for above binary number is

1 1 0 1 1 1 0



1 1 0 0 1 1 0 1 0

-2 3 2

Hence $(1101110)_2$= (-2 3 2) $_{QSD}$

3.2 Steps for QSD Addition

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

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.

3.3 Rules for QSD Addition

Two rules to perform QSD addition in two steps has been stated to avoid further rippling of carry:

Rule 1: First rule states that the magnitude of the intermediate sum must be less than or equal to 2 i.e., it should be in the range of -2 to +2.

Rule 2: Second rule states that the magnitude of the intermediate carry must be less than or equal to 1 i.e., it should be in the range of -1 to +1.

Table II. shows the possible addition results of two QSD digits.

Table II. Possible addition results of two QSD digits

	-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

Since the QSD number system has redundancy feature, we choose only those QSD numbers which satisfy above mentioned two rules. Thus the mapping between input, addend and augend, output, intermediate sum and carry has been shown in Table III.

TABLE III. Intermediate Sum and Carry Representation

Sum	Possible 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	$12,2\bar{2}$	12

In second step of QSD addition, the intermediate sum of current digit (in the range of -2 to +2) is added with intermediate carry of lower significant bit (in the range of -1 to +1); hence the addition result doesn't go beyond 3 i.e. it is in the range of -3 to +3. Single digit QSD is required to represent this number and hence there is no further rippling of carry.

Example 1: To perform QSD addition of two numbers A=143, B=100 First both numbers are represented in equivalent QSD numbers.

$(143)_{10}$ = $(2\ 0\ 3\ 3)_4$ and $(100)_{10}$= $(1\ 2\ 1\ 0)_4$

A= (143)10 2 0 3 3

B= (100)10 1 2 1 0

Sum= 3 2 4 3

IC 1 0 1 1

IS $\bar{1}\ 2\ 0\ \bar{1}$

O/p $1\ \bar{1}\ 3\ 1\ \bar{1}$

The sum output is $(1\ \bar{1}\ 3\ 1\ \bar{1})_4$ which is equal to $(243)_{10}$

Example 2: To perform QSD subtraction of two numbers A=143, B= -100 First both numbers are represented in equivalent QSD numbers.

(143)10 = $(2\ 0\ 3\ 3)_4$ and (-100)10= $(\bar{1}\bar{2}\bar{1}\ 0)_4$

A= (143)10 2 0 3 3

B= (-100)10 $\bar{1}\bar{2}\bar{1}$ 0

Sum= $1\ \bar{2}\ 2\ 3$

IC 0 0 0 1

IS $1\ \bar{2}\ 2\ \bar{1}$

O/p $0\ 1\ \bar{2}\ 3\ \bar{1}$

The sum output is $(0\ 1\ \bar{2}\ 3\ \bar{1})_4$ which is equal to $(43)_{10}$.

From above examples it is clear that both addition and subtraction operations are performed without rippling of carry/borrow.

4 Results and Discussion

Fig. 3 Radix-4 multiplication result for unsigned number

Fig. 4 Radix-4 multiplication result for signed number

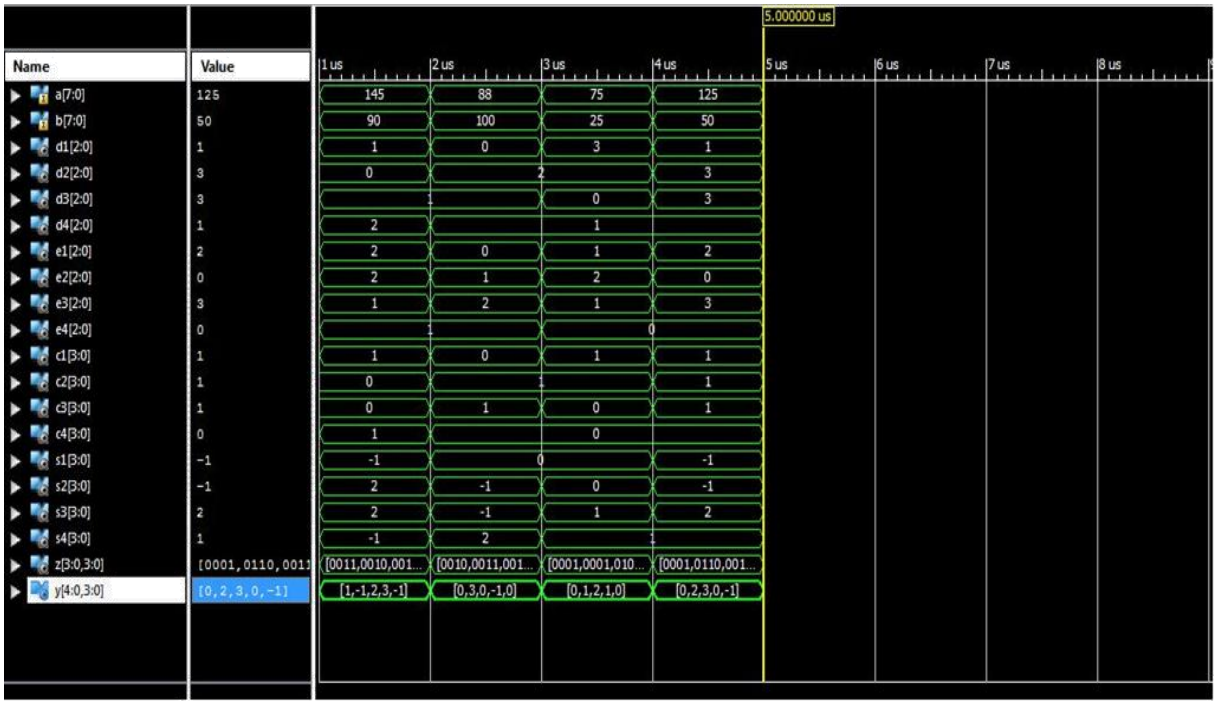


Fig. 5 Addition Result

Fig. 6 Subtraction Result

Table IV. Device utilization of Radix-4 booth multiplier

Logic Utilization	Used	Available	Utilization
Number of 4 input LUTs	169	7168	2%
Number of occupied slices	86	3584	2%
Number of slices containing only related logic	86	86	100%
Number of bonded IOBs	33	141	23%

Table V. Device utilization of QSD addition/subtraction for current work and previous work

	Current work			Previous work		
Logic Utilization	**Used**	**Available**	**Utilization**	**Used**	**Available**	**Utilization**
Number of slices	11	3584	0%	17	4104	0%
Number of 4 input LUTs	19	7168	0%	30	1312	2%
Number of bonded IOBs	36	141	25%	71	232	30%

The multiplication based on Radix-4 Booth algorithm and addition/subtraction operations have been simulated on ISim simulator of Xilinx 13.4 software for which above results are obtained. Table IV give device utilization information of Radix-4 Booth multiplication. The comparison of device utilization of addition/subtraction operations for current work and previous work has shown in Table V respectively.

5 Conclusion

The paper presents implementation of three basic arithmetic operations based on the radix-4 scheme. Radix-4 multiplication halves the number of partial products and acts as faster multiplication method. Another two basic arithmetic operations viz., addition/subtraction are implemented by using higher radix system i.e. QSD system for which radix is 4. The carry free and borrow free operations are performed due to redundancy feature of QSD, which results in faster computation.

Acknowledgments

I am sincerely thankful to my project guide Prof. Anil G. Sawant and Prof. V. S. Hendre**,** Head of our Department, for approving our project work with great interest.

References

[1] Snehal B. Sahastrabudhey, K.M. Bogawar, “Arithmetical Operations in Quaternary System Using VHDL”, IJCSET, Vol 2, Issue 4, pp. 1160-1163, April 2012.

[2] 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, Vol.2, Issue 5, May 2015.

[3] S. JakeerHussain, K. SreenivasaRao, "Design and implementation of Fast Addition Using QSD for Signed and Unsigned Numbers", International Journal of Engineering Research, Volume No. 3, Issue No: Special2, pp: 52-54, March 2014.
[4] 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.
[5] G. Manasa, M. DamodharRao, 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.
[6] C. V. Sathish Kumar, P. Jaya Rami Reddy, "Implementation of Fast Adder Using QSD for Signed and Unsigned Numbers", International Journal of Science, Engineering and Technology Research (IJSETR), Volume 3, Issue 11, November 2014.
[7] Kothuru. Ram Kumar, B. Praveen Kumar, "Fast Addition Using QSD VLSI Adder for Better Performance", International Journal of Trend in Research and Development, Volume 2(6), Nov-Dec 2015.
[8] SukhmeetKaur, Suman and Manpreet Signh Manna, "Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix 2)", Advance in Electronic and Electric Engineering, Vol. 3, No.6, pp.683-690, 2013.
[9] ShubhiShrivastva, PankajGulhane, "Optimized model of Radix-4 Booth Multiplier in VHDL", International Journal of Emerging Technology and Advanced Engineering, Vol.4, Issue 9, September 2014.
[10] V .R. Raut, P. R .Loya, "FPGA Implementation of Low Power Booth Multiplier Using Radix-4 Algorithm", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol.3, Issue 8, August 2014.
[11] K. Babulu, G. Parasuram, "FPGA Realization of Radix-4 Booth Multiplication Algorithm for High Speed Arithmetic Logics", International Journal of Computer Science and Information Technologies, vol.2 (5), 2011.
[12] Bodasingi Vijay Bhaskar, Valiveti Ravi Tejesvi, Reddi Surya PrakashRao, "Implementation of Radix-4 Multiplier with a parallel MAC unit using MBE Algorithm", International Journal of Advanced Research in Computer Engineering and Technology, vol.1, Issue 5,July2012.
[13] Wai-Leong Pang, Kah-Yoong Chan, Sew-Kin Wong, Choon-Siang Tan, "VHDL Modeling of Booth Radix-4 Floating Point Multiplier for VLSI Designer's Library" WSEAS TRANS. On SYSTEMS. Issue 12, Vol. 12, December 2013.
[14] Rashmi Ranjan, Pramodini Mohanty, "A New VLSI Architecture of Parallel Multiplier Based on Radix-4 Modified Booth Algorithm using VHDL", International Journal of Computer Science and Engineering Technology (IJCSET), vol.3, No.4, April 2012.
[15] Khalid Javeed, Xiaojun Wang, Mike Scott, "Serial and Parallel Interleaved Modular Multipliers on FPGA Platform", 2015 25th International conference on Field Programmable Logic and Applications (FPL), pp.1-4.
[16] S.ShafiullaBasha, Syed. Jahangir Badashah, "Design and Implementation of Radix-4 Based High Speed Multiplier using Minimal Partial Products", International Journal of Advances in Engineering & Technology, vol.4, Issue 1, pp.314-325, July 2012.
[17] A. B. Pawar, "Radix-2 Vs Radix-4 High Speed Multiplier", International Journal of Advanced Research in Computer Science and Software Engineering, Vol.5, Issue 3, March 2015.
[18] Sachin Dubey, Reena Rani, Saroj Kumari, Neelam 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 2013)
[19] Ameya N.Bankar, Shweta Hajare, "Design of Arithmetic Circuit using Quaternary Signed Digit Number System", International Conference on communication and signal processing, April 3-5, 2014, India.
[20] Loghman Rahimzadeh, Mohammad Eshghi, Somayeh Timarchi, "Radix-4 Implementation of Redundant Interleaved Modular Multiplication on FPGA", The 22nd Iranian Conference on Electrical Engineering (ICEE 2014), May 20-22, 2014.
[21] Shubhi Shrivastva, PankajGulhane, "Comparative Analysis on Power and Delay Optimization of Various Multipliers using VHDL ", International Journal of Electrical and Electronics Research, Vol.2, Issue 3, pp: (182-191), July-September 2014.
[22] Reena Rani, Laxmi Kant Singh, Neelam Sharma, "FPGA Implementation of Fast Adders using Quaternary Signed Digit Number System", 2009 International Conference on Emerging Trends in Electronic and Photonic Devices & Systems (ELECTRO 2009