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, Quaternary signed digit (QSD) numbers whose radix is 4 are used in arithmetic operations to achieve the carry free arithmetic operations. The range of QSD number is from -3 to 3. In any n digit QSD number, each digit can be represented by a number from the digit set \{-3,-2,-1,0,1,2,3\}. Carry free addition and other arithmetic operations on large number of digits such as 64, 128, or more can be implemented with the fixed constant delay and less complexity. Modelsim SE-6.3f and Xilinx ISE-9.1i softwares are used for simulation of design of QSD adder system. Design is implemented using VHDL language and synthesized using Xilinx ISE-9.1i with Spartan 3 Field programmable Gate array (FPGA) Starter kit board.

Keywords - Carry free addition, Fast computing, QSD(Quaternary signed Digit),VHDL,VLSI.

I. INTRODUCTION

There are various applications in the field of digital system such as computers, process controllers, signal processors, computer graphics, image processing, optical computing in the optoelectronic devices such as lasers, array illuminators, SEED(self electro-optic effect devices), spatial light modulators in which high speed arithmetic operation became significant due to the spread of wireless communication and portable computing system.

The organization of paper is as follows: In this paper, we review some of the work related to QSD adder system. Section II explains the Quaternary signed digit numbers. Section III explains the converter of Decimal number to Quaternary signed digit number system and their simulation results. Section IV explains the converter of Quaternary signed digit number to decimal number system and their simulation results. Conclusion part of the paper is given in section V. In this paper, we explain the the discussion of work of previous research which belongs to different adders related to the proposed work as follows. As the name implies, carry look ahead adder [1] predicts the block carry output instead of calculating carry bit by bit. The CLA block defines two different terms, propagate and generate. The delay of a carry look ahead adder is proportional to N/K where K is the number of bits per block. As the number of bits per block increases, the size of generate circuits increases exponentially. Therefore larger block size would require more area and power. As the operand size or number of bits for addition increases, computation time also increases which lead to the slow speed of machine because addition is performed in a time proportional to log 2 n. Redundant binary signed digit numbers [2] had been utilized as one of the SD representation proposed by Avizienis for high speed VLSI multiplication. It has a fixed radix 2 and a digit set {1,0,1} where 1 denotes -1. An n digit redundant binary integer \( X = [x_{n-1} \ldots x_0]_{SD2} \) (\( X \in \{1,0,1\} \)) has the value \( \sum_{i=0}^{n-1} x_i \times 2^i \). It is similar to an unsigned binary integer except that every xi can be 1. The redundant binary representation allows the existence of higher redundancy. Since a high level redundancy can be too costly, in BSD number there are several ways to represent an integer. Owing to the redundancy, it suffers from limited carry propagation delay with somewhat more complex addition process and more computation rule used in the algorithm. Also it leads to the increased circuit complexity which incorporates larger number of cascaded gates. Consequently, BSD adder takes more space to store the...
number compared to that of proposed work of QSD adder system. In the implementation of Quaternary Half adder and Full adder[3], two bit natural representation of binary logic is used for each Quaternary number and addition is performed in binary itself. Two Quaternary to Binary converters, sum generator, carry generator, two Binary to Quaternary converter in case of half adder and two encoder ,two code generator, sum and carry blocks in case of Full adder were designed in this implementation. Simulation of these circuits is carried out targeted for 180nm technology using Hspice, and cosmos tools. Although these circuits shows high performance, these recent advances in technologies for integrated circuits make large scale arithmetic circuits for VLSI implementation[2][3]. However, arithmetic operations still suffer from known problems including limited number of bits, propagation time delay and circuit complexities. In this paper, we propose a high speed QSD arithmetic logic design unit which is capable of carry free addition, borrow free subtraction, division, up-down count multiplications. A fixed number of minterms for any word length of operand are employed by this QSD addition/subtraction.The optimized design of QSD Adder/QSD subtractor can be made in order to obtain the constant delay for any operand size and perform the high speed arithmetic operation with less cascaded gates and also it uses sufficient redundancy since it takes less space than that of BSD adder system to store the number. In this system carry propagation chains are eliminated which reduce the computation time substantially, thus increasing the speed of machine.

II. QUATERNARY SIGNED DIGIT NUMBERS

QSD numbers are represented using 3 bit 2’s complement notation. Each number can be represented by:

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

where Xi can be any value from the set \{3,2,1,0,1,2,3\} for producing an appropriate decimal representation. A QSD negative number is the QSD complement of QSD positive number i.e.\(3 = -3,2 = -2,1 = -1\). For digital implementation large number of digits such as 64, 128, or more can be implemented with fixed delay. A high speed and area effective adders and multipliers can be implemented using this number system. Also we can obtain redundant multiple representation of any integer Quantity using this QSD number system. Examples of n digit QSD number are as follows: 2310,21301023,3010230231230322 etc.

III. DECIMAL NUMBER TO QSD NUMBER CONVERTER

We review some ideas related to this decimal number to QSD number converter based on the text book[4]. We can achieve making of such kind of algorithm which aids to convert any n digit decimal number into higher radix QSD number system. For the same purpose we exploit some fundamentals as we are most familiar with those are as given below, and these fundamental plays an important role in corresponding algorithm. The numbers that are positive are called unsigned, and numbers that can also be negative are called signed. Here, two types of binary numbers are defined.1. Unsigned binary number consist of only magnitude and this number is always positive.2. Signed binary number consists of magnitude as well as sign. We can use both unsigned binary integer and signed binary integer depending on the requirement of inputs given to algorithm. In the decimal number system the sign of a number is indicated by a + or − symbol to the left of the most- significant digit. In binary number system the sign of a number is denoted by the left most bit. The left-most bit is equal to 0 for a positive number and for a negative number it is equal to 1. Therefore, in signed numbers the left most bit represents the sign, and the remaining n-1 bits represent the magnitude, as illustrated in fig 1. It is important to note the difference in the location of most-significant bit (MSB). In unsigned numbers the magnitude of a number is represented by all bits; hence all n bits are significant in defining the magnitude. Therefore, the MSB is the left-most bit, \(b_{n-1}\). Since the left most bit of unsigned integer is called the MSB but left most bit of signed integer is not MSB and in signed numbers there are n-1 significants bits, and the MSB is in bit position \(b_{n-2}\) as illustrated in fig a and fig b respectively.

This algorithm can be directly applied to both unsigned and 2’s complement binary integer addition/subtraction. The algorithm depicts that decimal number is given as input in the form of n bit binary number as modelsim and Xilinx softwares takes the inputs in binary form. Algorithm states that it takes any n digit decimal number and converts into its equivalent Quaternary signed digit number whereas the given input is positive decimal number or negative number. If decimal number is positive, then after conversion we get QSD number in which each digit is positive is found, and also each digit is represented with 3 bit 2’s complement notation is displayed in simulation result. If decimal number is negative, then after conversion we get QSD number in which each digit is negative is found and also each digit is represented with 3 bit 2’s complement notation is displayed in simulation result.

The conversion of a decimal number into QSD number can be performed by successively dividing the decimal number by 4 and keeping the track of remainders as follows: Suppose that a decimal \( D = d_{k-1} \ldots d_1d_0 \), with a value \( V \), is to be converted into QSD number \( \text{QSD} = qsd_{n-1} \ldots qsd_2qsd_1qsd_0 \). Thus

\[
V = qsd_{n-1} \times 4^{n-1} + \ldots + qsd_2 \times 4^2 + qsd_1 \times 4 + qsd_0
\]

If we divide \( V \) by 4, the result is

\[
\frac{V}{4} = qsd_{n-1} \times 4^{n-2} + \ldots + qsd_2 \times 4^1 + qsd_1 + \frac{qsd_0}{2}
\]

The quotient of this integer division is

\[
qsd_{n-1} \times 4^{n-2} + \ldots + qsd_2 \times 4 + qsd_1,
\]

and the remainder is \( qsd_0 \). If the remainder is 0, then \( qsd_0 = 0 \); if it is 1, then \( qsd_0 = 1 \); if it is 2, then \( qsd_0 = 2 \); if it is 3, then \( qsd_0 = 3 \); if it is -1, then \( qsd_0 = -1 \); if it is -2, then \( qsd_0 = -2 \); if it is -3, then \( qsd_0 = -3 \). Observe that the quotient is just another QSD number, which comprises \( n-1 \) digits, rather than \( n \) digits. Further dividing this number by 4 yields the remainder \( qsd_1 \). The new quotient is

\[
qsd_{n-1} \times 4^{n-3} + \ldots + qsd_2 \times 4 + qsd_1
\]

Continuing the process of dividing the new quotient by 4, and determining one digit in each step will produce all digits of the QSD integer. The process continues until the quotient becomes 0. Note that the least-significant digit (LSD) is generated first and the most significant digit (MSD) is generated last. These remainders are nothing but the collection of digits which constitute QSD number. Here simulation results of two examples of conversion of decimal number to QSD number are illustrated in figure 2 and figure 3.

**Ex.1**
\( (127)_{10} = (1333)_{\text{qsd}} \)

**Ex.2**
\( (32768)_{10} = (0000000000000000)_{\text{qsd}} \)
Using 2’s complement representation, an n-bit number $B = b_{n-1} \ldots b_1 b_0$ represents value $V(B) = (-b_{n-1} \times 2^{n-1}) + b_{n-2} \times 2^{n-2} + \ldots + b_1 \times 2^1 + b_0 \times 2^0$. The largest negative number, 100..00, has the value $-2^{n-1}$. The largest positive number, 0111..11, has the value $2^{n-1} - 1$.

If the decimal number is 0 as input in the form of binary number then output is always 0. It is noticed that 0 is neither negative nor positive.

**IV. QSD NUMBER TO DECIMAL NUMBER CONVERTER**

Any n digit QSD number can be converted into a decimal number by using the equation as follows:

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

where $X_i$ can be any value from the set {0, 1, 2, 3} for producing an appropriate decimal representation. In this algorithm we use QSD number, or base 4 system in which digits can be 0, 1, 2, 3 which are represented in 3-bit 2’s complement notation. According to this above equation we use positional number representation which depicts that powers of 4 with the QSD number are implied by the position of digits, however each digit represents a multiple power of 4. In general, QSD integer is expressed by an n-tuple comprising n QSD digits QSD = $q_{sd_{n-1}}q_{sd_{n-2}}\ldots q_{sd_1}q_{sd_0}$ which represents the value

$$V(QSD) = q_{sd_{n-1}} \times 4^{n-1} + q_{sd_{n-2}} \times 4^{n-2} + \ldots + q_{sd_1} \times 4^1 + q_{sd_0} \times 4^0$$

In this manner, a QSD number is converted into a decimal number simply by applying above equation and evaluating it using decimal arithmetic. Some examples are given below:

Ex.1 $(10\overline{2}1)_{QSD} = 1 \times 4^3 + 0 \times 4^2 + \overline{2} \times 4^1 + 1 \times 4^0 = 64 + 0 - 8 + 1 = (57)_{10}$

Ex.2 $(7027)_{QSD} = (-57)_{10}$

Ex.3 $(13333333)_{QSD} = (32767)_{10}$

(-20000000)$_{QSD}$ and $(13333333)_{QSD}$ both these numbers represent largest number in the range from (-32768) to (32767). Beyond these range we can get decimal numbers from any n digit QSD numbers. Simulation results for above examples are illustrated in figure 4, figure 5, figure 6.
V. CONCLUSION

In this paper, we discussed the VHDL implementation of both the converters decimal number to QSD number converter and QSD number to decimal number converter regarding the n digit QSD numbers and decimal numbers having the ranges from \(-8\) to \(+7\), \(-128\) to \(+127\), \(-32768\) to \(32767\) etc. those can be represented with 4bit, 8 bit, 16 bit etc. respectively in 2's complement form. Also it is concluded that largest n digit QSD number uses largest range of decimal number beyond the above range. Because decimal numbers having largest range requires large numbers of bits to represent those numbers in its 2’s complement binary form.

REFERENCES


