Logic Design of Elementary Functional Operators in Quaternary Algebra

Asif Faiyaz, Sarah Nahar Chowdhury, and Khandakar Mohammad Ishtiak

***Abstract*—Multivalued logic is an extension of Boolean algebra with high radix approaches and is preferable over conventional binary logic operations for reduction in interconnection cost, chip area both on-chip and between chips and high information handing capability. This paper includes the design of elementary combinational quaternary operators that have sufficient representative capability to efficiently implement in intricate quaternary arithmetic circuits. Design of several combinational logic circuits have been presented which can function individually and in logic blocks for designing further complex circuits resulting in a reduction of circuit complexity and better speed processing in integrated circuit technology.**

***Index Terms*—Cycle gate, max and min gate, multivalued logic, quaternary algebra.**

I. Introduction

Binary logic has been predominant in embedded system and a fundamental for computer programming and mathematical logic due to its easy accessibility and widespread use in logic circuits. Operating with binary logic implies to controlling the real world with computers and that an alternative improved approach with a better usage of transmission path, storage and processing of large amount of information in digital signal processing even exist seems somewhat impossible. Yet, Moore's law states that, the number of transistors on integrated circuits doubles approximately every two years. As this law has been by far proven accurate, it is high time we considered alternative approaches to reduce this tremendous positive rate of elements used in integrated circuits. It has been suggested that by increasing the processing speed, memory capacity, sensors or memory states, this significant rate can be inhibited to an exponential rate. But, due to the inherent limitation data representation system of only two distinct levels {0, 1}, binary logic impedes the scope for multiple states and lacks high speed and information handling capacity. On the other hand, multivalued number systems, such as ternary and quaternary systems with a radix more than '2' (p>2) emerges with the immediate benefit of larger information handling and storage capacities.

Multivalued logic system introduces new operators in addition to binary values {0, 1} and is a proposed extension of the idea that n valued logic can be used instead of two logical values (that is, true or false, logic high or low) where n>2 [1].

Perhaps one of the most tangible immediate benefits of higher-radix approaches like quaternary logic lies in their potential for reduction of the wiring congestion and interconnection cost [2]. Using a single conductor to transmit three or more discrete voltage or current values allows for greater information content per wire and thus results in a circuit with reduced conductors and logic gates than the binary-valued counterpart. Furthermore modeling of many complex logic systems and associated algebras are possible due to the exponentially increasing number of operators with respect to the cardinality of the multiple logic values.

Although multivalued logic particularly quaternary algebra is not as widespread as binary, it has gained much appreciation in recent years due to its higher information handling capacity of much more complex algorithms in emerging topics like optical and quantum computing [1]. That is why it is prerequisite to express its fundamental operators in terms of equations and logic diagram for further implementation in multiplex algorithms. In this paper, we have designed some important functions like max, min, mod-sum, mod-difference, successor, predecessor, 2^{nd} level successor or predecessor and 3^{rd} level successor and predecessor using earlier proposed logic functions and basic operators proposed in [3]-[6] which can be implemented subsequently in composite circuits.

II. Quaternary Algebra

Multivalued logic (MVL) or nonbinary-valued system utilizes variables that can take on a discrete set of values with cardinality of $n \geq 3$ and quaternary algebra which has been derived as propositional or quantum logic from MVL algebra is defined over four finite sets of logic values [1]. Hence, while multi valued logic deals with infinite number of values as discrete variables, quaternary algebra is based upon the discrete variables {0, 1, 2, 3} including the binary values {0, 1}. Quaternary states {0, 1, 2, 3} with a set of operators and axioms are used to define quaternary algebra. Each of the quaternary states {0, 1, 2, 3} has its two bits binary equivalents 00 (absolute low), 01 (intermediate low), 10 (intermediate high) and 11 (absolute high) and each of the quaternary bits is called 'qudit' [3], [7], when expressed in numbers. The logic values can also be indicated by two binary digits 'a1' and 'a0', respectively which is inscribed and packed together using the following notion $A=\{a1, a0\}$ where the term '$2a1 + a0$' denotes the magnitude of the variable 'A' in decimal system [4]. Quaternary states are sub categorized into symmetrical and asymmetrical based upon their position of bits. If the bits of the binary equivalent of quaternary states interchange their position and still the quaternary states remain unchanged then they are known as symmetrical.

Manuscript received September 9, 2014; revised January 5, 2015.

The authors were with the Ahsanullah University of Science and Technology, Dhaka, Bangladesh (e-mail: asif.faiyaz@gmail.com).

Absolute states (0, 3) are symmetrical as the change of bits in binary equivalent does not change the corresponding quaternary value. If the transposition of position of bits changes the corresponding binary value then they are known as asymmetrical. Intermediate states (1, 2) are asymmetrical as interchanging the binary equivalents changes the quaternary state 1 to 2 and vice versa [6].

Quaternary algebra is in congruence with binary logic in terms of the basic operators like OR, AND, BUFFER, BASIC INVERTER, XOR, BASIC NAND, BASIC NOR and BASIC XNOR and that is why the interfacing of binary to quaternary can easily be conducted by just using an encoder. So, quaternary algebra can be used as models for the initial design of logic circuits whether they are implemented with MVL signal levels or binary after being encoded. Again, using the mainstream fundamental operators, the functional operators which are analogous to binary logic can easily be derived and all the logic blocks of quaternary are compatible with binary logic design making it a paragon for dual purpose.

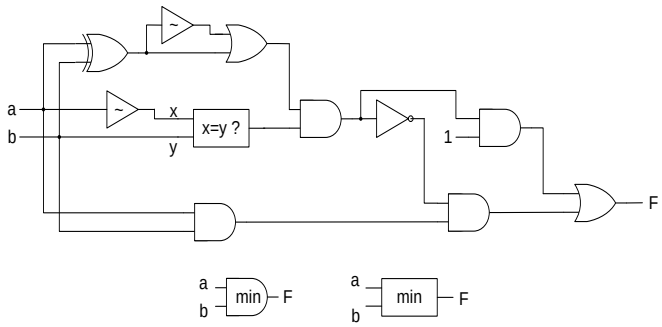

Fig. 1. Design of Min gate using basic gates.

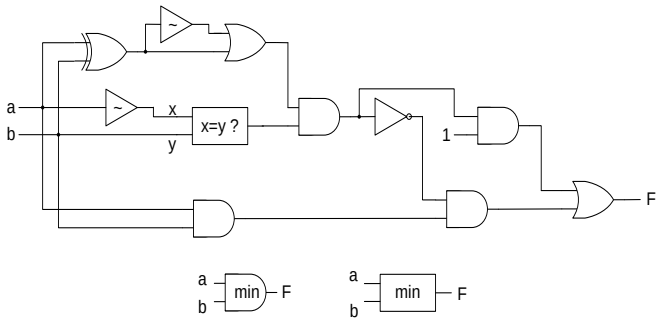

Fig. 2. Design of Max gate using basic gates.

III. Proposed Idea

Just like Boolean algebra, it is necessary to provide an effective logic design and well-defined framework for expressing and manipulating functions in quaternary algebra. Combinational circuit is a vital element for any digital system. In many applications, that is why, it is imperative that the operators of the algebra have simple and efficient circuit implementations for reducing circuit complexity. Although functions like max, min, cycle gate (forward and reverse cycle), successor and predecessor, 2nd level successor and predecessor have already been introduced [3], [4], yet, no specifications to the truth table, logic diagram and corresponding equations have been stated so far abating the process of further implementation of these functions in complex circuitry. The paper proposes the techniques for finding the truth table, logic diagram and the equation for some consequential functional operators like max, min, cycle gate (forward and reverse cycle), successor and predecessor, 2nd level successor and predecessor which can be further implemented to solve circuit complexity. Henceforth, our proposed quaternary logic designs can be integrated efficiently for designing some conventional circuits.

A. Min and Max Gate

Seeing in Fig. 1 and Fig. 2. The Min function is used to compare the minimum among the several literals [3]. A representation of a quaternary switching function that is analogous (but not identical) to the binary sum of minterms representation can be formulated using the Max function in place of the binary inclusive OR function and product terms may be formed using the Min function in place of the binary AND operation. But after obtaining the truth table of Min and comparing it with the corresponding truth table for AND function, it can be concluded that the Min function is similar to AND function with the exception for the particular case of inputs '1' and '2' or '2' and '1' [4] (see Table I). Similarly, the comparison between Max and OR operators also shows resemblance in their properties and output except when the inputs are '1' and '2' or '2' and '1' [4]. Many of the existing quaternary logic schemes have min and max as operators and these operators have already been realized physically [8]. But the lack of representation in logic design makes it difficult to implement these functions in complex circuits. So, we have proposed a new set of equations congruous with the established truth table along with their respective complementary logic design which are of their simplest form and are further verified using matlab. Min gate function can be represented by the following equation.

$$\text{Min}(A, B) = \begin{cases} 1 \ ; & when \quad A = 1 \ and \ B = 2 \\ & or \quad A = 2 \ and \ B = 1 \\ A.B \ ; & otherwise \end{cases}$$

Clearly we can understand that while designing the logic diagram we cannot replace min function with AND gate rather we have to undergo additional changes to obtain the output for (1, 2) and (2, 1). For designing the logic diagram therefore we have utilized the concepts of special gates, different theorems and properties of quaternary algebra. Initially in order to just obtain (1, 2) and (2, 1) we have utilized the concept of bitswap and equity functions. We know, bitswap operator interchanges the asymmetric inputs keeping the symmetric unchanged and equity function provides the output '3' for the similar inputs and rest '0' [4]. So together we obtain '3' for (0, 0), (1, 2), (2, 1) (3, 3) and '0' otherwise. In the very next step, we need to design a function which separates the inputs (0, 0) and (3, 3) from the other two sets. Observing the pattern, we have introduced XOR operator here which provides '0' for similar inputs and 'greater than 1' for dissimilarity. Although after using this function, we have separated the desired functions yet we lose the output '3' for inputs (1, 2) and (2, 1). So, in order to obtain this output we have utilized one of the

properties of quaternary algebra which is –

$$\sim a + a = \begin{cases} 3\ ; & a \neq 0 \\ 0\ ; & a = 0 \end{cases}$$

We have focused on the two outputs '3' and '0' for utilizing the property 'a.3 = a' and 'a.0 = 0' of AND gate. As a result, we have placed an AND gate at the next step in order to obtain an output '3' for only inputs (1, 2) and (2, 1) and '0' otherwise. We then place an inverter connected to an AND gate to ensure an output of '3' for any combination of input apart from (1, 2) and (2, 1) for which it is '0'. After this very step we need to convert the output '0' for inputs (1, 2) and (2, 1) to '1' keeping the rest unchanged by utilizing the property,

$$a.1 = \begin{cases} 0\ ; & a = 0 \\ 1\ ; & a = 3 \end{cases}$$

Placing it before the inverter we obtain the desired result which is then connected to an OR gate to provide the required output. The logic design of min function can be manipulated into different ways through different combinations of gates, yet we managed to ensure a novel design with minimum gates after going through different combinations.

We can also observe the equations of Max gate derived from perceiving the similarity of this function with other basic functions and the truth table.

$$\text{Max}(A, B) = \begin{cases} 2 \ ; & when \quad A = 1 \ and \ B = 2 \\ & or \quad A = 2 \ and \ B = 1 \\ A + B \ ; & otherwise \end{cases}$$

In case of max function similarly, while comparing with OR function we observe similarity except for (1, 2) and (2, 1). As a result we use the same procedure stated above to separate the contradictory functions from the rest and then using the properties of quaternary algebra to obtain the required result.

TABLE I: TRUTH TABLE FOR MAX FUNCTION, MIN FUNCTION, AND FUNCTION AND OR FUNCTION

Input		Output			
B	**A**	**AND**	**MIN**	**OR**	**MAX**
0	0	0	0	0	0
0	1	0	0	1	1
0	2	0	0	2	2
0	3	0	0	3	3
1	0	0	0	1	1
1	1	1	1	1	1
1	2	0	1	3	2
1	3	1	1	3	3
2	0	0	0	2	2
2	1	0	1	3	2
2	2	2	2	2	2
2	3	2	2	3	3
3	0	0	0	3	3
3	1	1	1	3	3
3	2	2	2	3	3
3	3	3	3	3	3

B. Forward Cycle and Reverse Cycle Gate

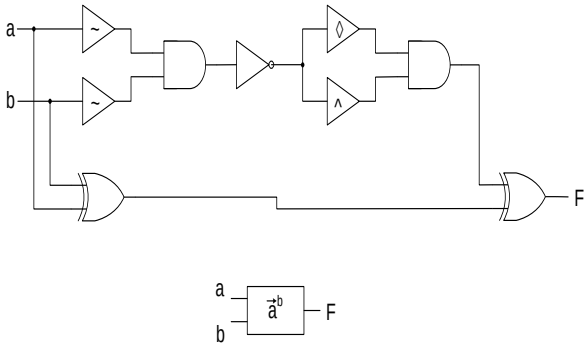

Fig. 3. Design of Forward cycle gate using basic gates.

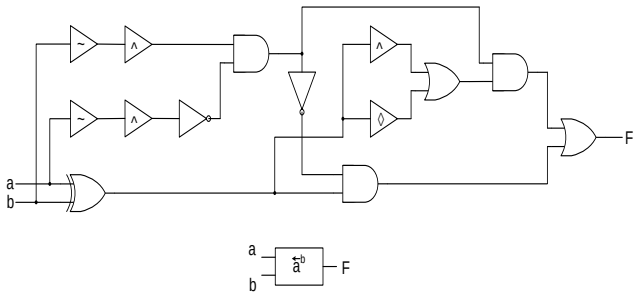

Fig. 4. Design of Reverse cycle gate using basic gates.

TABLE II: TRUTH TABLE FOR FORWARD CYCLE FUNCTION, REVERSE CYCLE FUNCTION AND XOR FUNCTION

Input		Output		
B	**A**	**XOR**	**Forward Cycle**	**Reverse Cycle**
0	0	0	0	0
0	1	1	1	1
0	2	2	2	2
0	3	3	3	3
1	0	1	1	3
1	1	0	2	0
1	2	3	3	1
1	3	2	0	2
2	0	2	2	2
2	1	3	3	3
2	2	0	0	0
2	3	1	1	1
3	0	3	3	1
3	1	2	0	2
3	2	1	1	3
3	3	0	2	0

Seeing in Fig. 3 and Fig. 4, Forward cycle or mod-sum and reverse cycle or mod-difference are cycle gates where the output varies taking the reference point as one input and varying with respect to the quaternary value of the other input [3]. In case of forward cycle gate, with respect to one reference point the movement of the input quaternary value is clockwise and in case of reverse gate it is anticlockwise. For example, for the two input values as 1 and 3, if the reference value is 1, then the output after moving in a clockwise direction by shifting twice becomes 0. On the contrary, for the reverse cycle gate the output after moving in an anticlockwise direction becomes 2. The following equation is derived and the logic diagram is further designed by observing the truth

table constructed by calculating the logical output and then verified using matlab.

$$\text{Forward cycle } (A, B) = \begin{cases} 0 \;;\; when \quad A=1 \;\; and \;\; B=3 \\ \qquad\quad or \quad A=3 \;\; and \;\; B=1 \\ 2 \;;\; when \quad A=B=1 \\ \qquad\quad or \quad A=B=3 \\ A \oplus B \;;\; otherwise \end{cases}$$

Observing the truth table of forward cycle (see Table II), we conclude the similarity between forward cycle and XOR gate except for inputs (1, 1), (1, 3), (3, 1) and (3, 3). As a result while designing the logic diagram apart from placing a XOR gate we need to redesign our circuit in order to obtain the required output for the exceptions. Observing the output we can speculate that the antonymous output varies from '0' to '2' and vice versa. Utilizing this understanding we have placed two BITSWAP gates in order to change the exceptional asymmetric input '1' to '2' and further placing an AND gate which provides an output (>=2) for the exceptional inputs and output (<=1) otherwise. After that, using an INVERTER and the following function,

$$a'\hat{a} = \begin{cases} 0 \;; & a > 1 \\ 2 \;; & a < 2 \end{cases}$$

We obtain an output of '2' for the exceptional inputs and '0' otherwise. Then using just an XOR gate with this particular combination we attain the desired output. Similarly, the Reverse cycle gate equation being derived is:

$$\text{Reverse cycle } (A, B) = \begin{cases} 1 \;;\; when \quad A=2 \;and\; B=1 \\ \qquad\quad or \quad A=0 \;and\; B=3 \\ 3 \;;\; when \quad A=0 \;and\; B=1 \\ \qquad\quad or \quad A=2 \;and\; B=3 \\ A \oplus B \;;\; otherwise \end{cases}$$

Observing the truth table of the Reverse cycle gate and comparing the output with that of XOR, we have used different combinations and properties to obtain the corresponding logic diagram.

C. Successor and Predecessor Gate

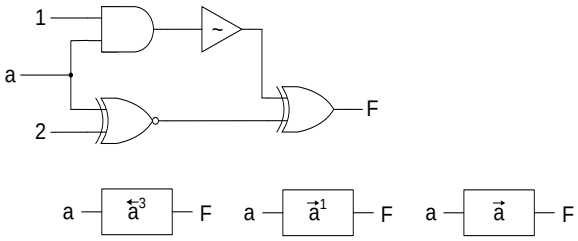

Fig. 5. Design of Successor gate using basic gates.

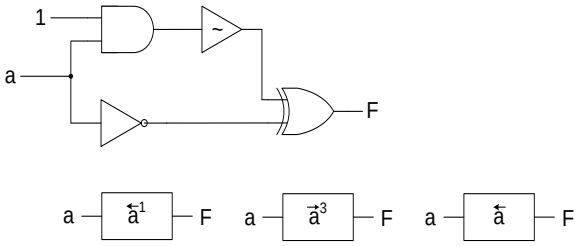

Fig. 6. Design of Predecessor gate using basic gates.

Seeing in Fig. 5 and Fig. 6, While functions like forward and reverse cycle shows us the interdependence of the quaternary inputs, functions like successor and predecessor give us the desire output as per the number of times the input shifts being fixed beforehand [3]. Predecessor or 3rd level successor gates give the preceding value or the output after the input shifts three times in the clockwise direction for the particular value of input. For example, for a particular input of 1, we can obtain the precedent value 0 or by shifting the input three times in the clockwise direction. Similarly, using the successor or 3rd level predecessor gate, we can obtain the succeeding value which is 2 for the particular above mentioned case. The logic diagrams are designed again by comparing the output with different properties of quaternary operators and applying different combinations of gates.

$$\text{Successor gate } (A) = \sim((A.1) \oplus \overline{(A \oplus 2)}$$

$$\text{Predecessor gate } (A) = \sim((A.1) \oplus \overline{A}$$

TABLE III: TRUTH TABLE FOR SUCCESSOR FUNCTION, PREDECESSOR FUNCTION, 2ND LEVEL SUCCESSOR OR 2ND LEVEL PREDECESSOR FUNCTION, 3RD LEVEL SUCCESSOR FUNCTION AND 3RD LEVEL PREDECESSOR FUNCTION

Input	**Output**				
A	**Successor**	**Predecessor**	**2nd level successor or 2nd level predecessor**	**3rd level successor**	**3rd level predecessor**
0	1	3	2	3	1
1	2	0	3	0	2
2	3	1	0	1	3
3	0	2	1	2	0

D. 2nd Level Successor or 2nd Level Predecessor Gate

Seeing Fig. 7, Just like Successor or Predecessor gate, using 2nd level successor or 2nd level predecessor gate, we can obtain the corresponding 2nd preceding and succeeding values after the input being shifted twice. We can conclude that using different levels of successor and predecessor gates we can obtain any desired value at any position which can be of immense benefit while designing complex logic circuits. The equation for this particular case has been derived as follows:

$$2^{nd} \text{ level successor gate} =$$
$$2^{nd} \text{ level predecessor gate} = (A \oplus 2)$$

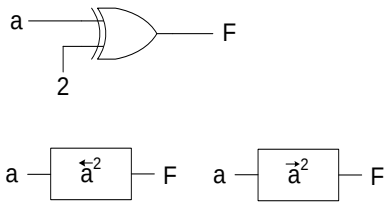

Fig. 7. Design of 2nd Level Successor or 2nd Level Predecessor gate using basic gates.

IV. Future Works

Although the corresponding equation and logic diagram derived for the functional operators can be easily interfaced and has been minimized using different combinations, further work has to be done to make it more compatible by deriving different theorems and properties as the central concept of functional completeness of quaternary algebra is still the underlying concept to be considered. The combinational circuits designed in this paper can be effectuated for designing further complex circuits like adder, multiplier, subtractor, multiplexer, comparator, flip-flops and other elementary sequential circuits. Many of the more recently approved algebras have been developed for the purpose of modeling MVL circuits based upon particular electronic components to be used as primitive circuit elements in their construction and the logic design of elementary functional operators brings us one step closer to physical implementation of quaternary circuits in the high performance microprocessor.

V. Conclusion

In this paper, we have constructed the logic design and respective equations of foundational functional operators and tested each derived equation using matlab. Despite the inconsistency of output of the fundamental operators in quaternary algebra, we have managed to formulate novel designs for relevant functions like max and min functions which are considered the framework of quaternary algebra because of their implementation as a set of max, min and equality operators which is sufficient enough to express any quaternary function algebraically. We have also depicted operators like predecessor and successor through their logic diagram which have historically been considered as distinct since they have direct circuit implementations and are particular cases of general cycle operation. Therefore, for the significant efficacy of these operators, new algebraic and logic design techniques have been developed in this paper which can be utilized in some future novel technology or implementation. Although binary logic can be professed as the pivot of controlling the embedded fundamental and digital system of modern times, yet quaternary algebra imposes some features which can provide immense benefit to the VLSI and quantum technology contributing to the design of novel electron devices like Carbon Nanotube Transistor, FinFET, G4-FET, Silicon Nanowire FET, etc. where quaternary logic is preferable rather than binary for fast processing.

References

[1] S Hurst, "Multiple-valued logic — Its status and its future," *IEEE Transactions on Computers*, vol. C-33, no. 12, pp. 1160-1179, 1984.
[2] A. M. H. Khan, "Reversible realization of quaternary decoder, multiplexer, and demultiplexer circuits," *Engineering Letters*, vol. 15, no. 2, pp. 203-207, 2007.
[3] D. M. Miller and M. A. Thorton, "Multiple valued logic: Concepts and representations," *Synthesis Lectures on Digital Circuits and Systems*, Morgan & Claypool Publishers, 2007.
[4] I. Jahangir, A. Das, and M. Hasan, "Formulation and development of a novel quaternary Algebra," *arXiv Preprint arXiv: 1108.5497*, 2011.
[5] A. Das, I. Jahangir, M. Hasan, and S. Hossain, "On the design and analysis of quaternary serial and parallel adders," in *Proc. IEEE Region 10 Conference*, 2010, p. 1691.
[6] I. Jahangir, N. M. D. Hasan, S. Islam, N. A. Siddique, and M. M. Hasan, "Development of novel quaternary algebra with the design of some useful logic blocks," in *Proc. 12th International Conference on Computers and Information Technology*, 2009, p. 107.
[7] Y. A. Gaidhani and N. K. Monica, "Design of some useful logic blocks using quaternary Algebra," in *Proc. CEE 2011*, India, 2011.
[8] V. Patel and K. S. Gurumurthy, "Design of high performance quaternary adders," *International Journal of Computer Theory and Engineering*, vol. 2, no. 6, December 2010.

Asif Faiyaz is currently pursuing his bachelor degree in electrical and electronic engineering from Ahsanullah University of Science and Technology, Dhaka, Bangladesh. His research interests include signal processing, pattern recognition, nanotechnology, quaternary logic, microwave communication etc. He is currently a student member of the Institute of Electrical and Electronics Engineers (IEEE).

Sarah Nahar Chowdhury is an undergraduate student in electrical and electronic engineering of Ahsanullah University of Science and Technology, Bangladesh. She is currently working as the secretary of Engineering Students' Association of Bangladesh (ESAB). Her research interests include solar cells, renewable energy, organic transistors, nanotechnology and quaternary algebra. She is a student member of the Institute of Electrical and Electronics Engineers (IEEE) since 2012.

Khandakar Mohammad Ishtiak was born in Bangladesh in 1986. He graduated from the Electrical and Electronic Engineering Department of Ahsanullah University of Science and Technology (AUST). Currently he is working as an assistant professor of the EEE Department in AUST. He is a member of the Institute of Electrical and Electronics Engineers (IEEE). His research interests include nanotechnology, solar cells, quaternary algebra, signal processing and analysis.