ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 2, Issue 4, April 2015

Design of some special operators using Novel Quaternary logic system

Ashwini Raut[1], Darshana Chaware[2], Snehal Jawanjal[3], Shrikant Bhoyar[4]

Lecturer, Electronics Engineering, Rajiv Gandhi College of Engineering, Nagpur, India[1,2,3,4]

Abstract: Designing of some special quaternary using novel quaternary logic system is proposed here. This logic system is based on the extension of Boolean algebra Instead of conventional Fuzzy logic or Galois Field theory, we have defined here, the basic as well as special Quaternary operators and designed these operators using 180 nm technology.

Keywords: Quaternary algebra, quaternary logic, Inward, Outward, Bitswap, AND, OR, XOR**.**

I. INTRODUCTION

For many years digital system has been represented by binary numbers 0 and 1. Boolean algebra is used as the backbone of binary logic and hence it governs the design of all digital circuits. With the development of multi-gate and gate-alI-around devices like G4-FET, Silicon Nanowire FET, Carbon Nanotube FET, FinFET, etc., it is now possible to couple several binary inputs for faster processing. These novel devices can be used more effectively if we move beyond binary logic. Some multi-valued logic systems such as ternary and quaternary logic schemes have been developed and they are being experimented for a long time [1]. These logic systems are often derived from Fuzzy logic or Galois field theory [2]. It is clear that a binary system can be converted into a quaternary system by coupling each pair of consecutive bits of a binary bit stream to form a quaternary bit stream. Then the binary system can be replaced by an equivalent quaternary system. The quaternary system has several advantages over binary logic [3],[4]. Since it requires half number of digits than equivalent binary data, it is good for storage. In quaternary system more information is contained in a single digit, so it requires less parallel circuits to process same amount of data than that needed in binary system. Newer devices offer flexible yet sophisticated design opportunities which can be utilized more efficiently and meaningfully if quaternary system is adopted.

In [5],[6], a novel quaternary logic has been proposed which is close to Boolean algebra. It extends binary functions in quaternary, at the same time incorporates some new functions of its own. Extending Boolean algebra makes this logic system ideal for replacing binary logic as its logic blocks are compatible with their binary counterparts. In this work, we begin with a brief introduction to the elements of the new logic system and then we present the design of various quaternary operators

II. THE NOVEL QUATERNARY LOGIC

The quaternary logic we are proposing here consists of quaternary states or variables and operators. The quaternary states are 0 (absolute low), 1 (medium low), 2 (medium high) and 3 (absolute high), which can also be represented as 2-bit binary equivalents 00, 01, 10 and 11. If the bits of the binary equivalent interchange their positions and still the quaternary state remains unchanged, then it is said to have binary symmetry; otherwise it is asymmetric. Thus 0, 3 are symmetrical and 1, 2 are asymmetrical. When expressed as a number, a single quaternary digit is called a qudit. The basic quaternary operators are defined as bitwise binary operators working on the binary equivalents and they are obtained from Boolean algebra. These are or, and, basic inverter or basic not and xor. The word basic is used before the ordinary inverter to differentiate it from other special inverters that will be discussed later. Also there are some compound basic operators like basic nand, basic nor and basic xnor. Here we propose some special operators to facilitate the development of this novel quaternary algebra. They are all single input operators and very useful in designing complex logic circuits. These

A. Special Operators **:**

(a) Outward inverter or full inverter
(b) Inward inverter or half inverter
(c) Binary bitswap.

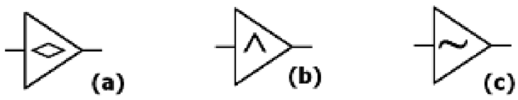

Fig. 1. Circuit symbols for the special quaternary operators (a) Inward inverter; (b) Outward inverter; (c) Binary bitswap.

The figure1 shows the Symbolic representation of outward inverter, inward inverter and binary bitswap. The outward inverter inverts the input just like the basic inverter, but after that it changes the asymmetrical values to nearest symmetrical values. It is also called as full inverter because it changes the output to its fullest value. That means the outward inverter will inverts 0, 1 into 3 and 2, 3 into 0.

The inward inverter inverts the input just like the basic inverter; but after that, unlike outward inverter, it changes the symmetrical values to nearest asymmetrical values. It

 DOI 10.17148/IARJSET.2015.2418

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 2, Issue 4, April 2015

is also called as half inverter because it pushes the output to its half value. That means the inward inverter will inverts 0, 1 into 2 and 2, 3 into 1.

The binary bitswap swaps the two bits of the binary equivalent of the quaternary operand. It leaves the symmetrical numbers unchanged but inverts (basic inversion) the asymmetrical numbers. That is why it is classified as a special inverter-like operator.

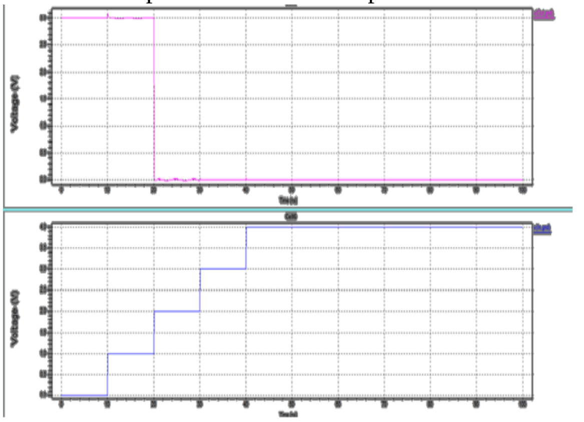

Fig2: Simulated input and output of outward inverter

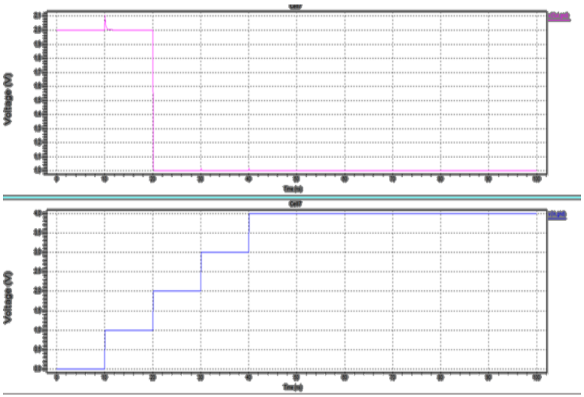

Fig3: Simulated input and output of inward inverter

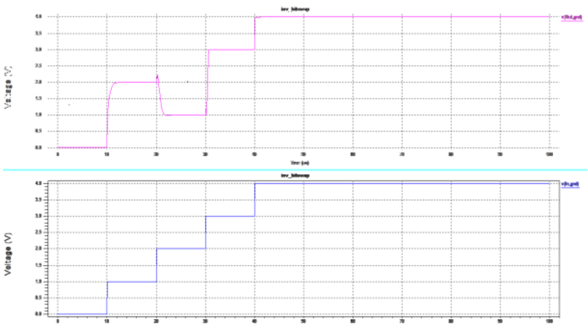

Fig4: Simulated input and output of binary bitswap

The basic inverter and binary bitswap can be decomposed into outward and inward inverters using the following expressions:

$$\overline{a} = \begin{cases} \hat{a} \ ; \ a \text{ symmetric} \\ a' \ ; \ a \text{ asymmetric} \end{cases} = \begin{cases} \hat{a} \ ; \ a = 0, 3 \\ a' \ ; \ a = 1, 2 \end{cases}$$

$$\tilde{a} = \begin{cases} a \ ; \ a \text{ symmetric} \\ a' \ ; \ a \text{ asymmetric} \end{cases} = \begin{cases} a \ ; \ a = 0, 3 \\ a' \ ; \ a = 1, 2 \end{cases}$$

TABLE I: QUATERNARY MULTI-INPUT OPERATORS

A	B	AND	OR	XOR	BASIC NAND	BASIC NOR	BASIC XNOR
0	0	0	0	0	3	3	3
0	1	0	1	1	3	2	2
0	2	0	2	2	3	1	1
0	3	0	3	3	3	0	0
1	1	1	1	0	2	2	3
1	2	0	3	3	3	0	0
1	3	1	3	2	2	0	1
2	2	2	2	0	1	1	3
2	3	2	3	1	1	0	2
3	3	3	3	0	0	0	3

Since inward and outward inverters can be made of simple CMOS inverter using multi-threshold technology [7], these expressions are useful to construct basic inverter and binary bitswap operator using such simpler circuits. It is anticipated that in near future, using novel devices like FinFET, G4-FET or SiNW-FET, it might be possible to implement basic inverter and binary bitswap without using their decomposed versions.

The special operators are necessary for designing logic circuits. Among them, binary bitswap is the most important one. It is used to define equality operator, which gives the sum of product (SOP) expression of a quaternary function.

B. COMPOUND DERIVATIVES OF SPECIAL OPERATOR

Like basic inverter, bitswap operator and special inverters can be cascaded with other operators to form compound operators such as inward nand, outward nor, bitswap xor.

(a) Inward NAND

Like basic inverter, if inward inverter, followed by basic AND operator, we get the inward NAND. The symbolic representation of inward NAND is shown in fig 5, with its simulated input output characteristic in fig 6.

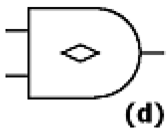

Fig5: Symbol of Inward NAND

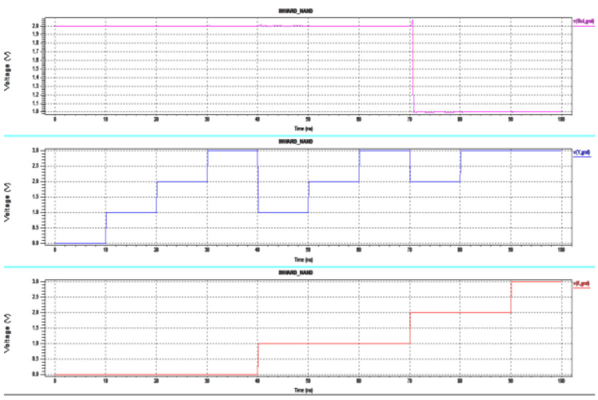

Fig6: Simulated input and output of inward NAND

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 2, Issue 4, April 2015

(b) Outward NAND

If outward inverter, followed by basic AND operator, we get the outward NAND. The symbolic representation of outward NAND is shown in fig 7, with its simulated input output characteristic in fig 8.

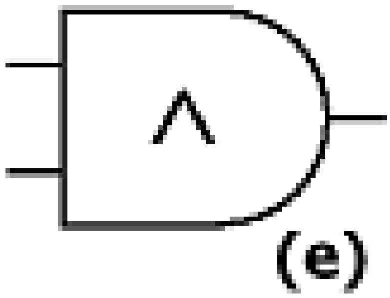

Fig7: Symbol of Outward NAND.

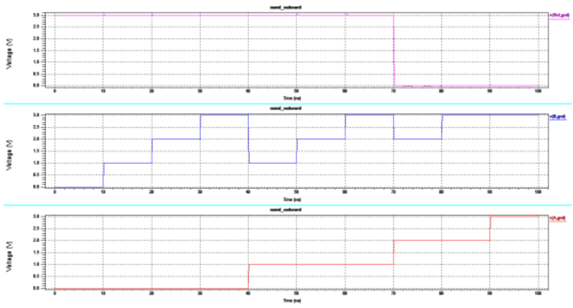

Fig8: Simulated input and output of outward NAND

(c) Bitswap AND

If the binary bitswap, followed by basic AND operator, we get the bitswap AND. the symbolic representation of bitswap AND is shown in fig 9, with its simulated input output characteristic in fig 10.

Fig9: Symbol of Bitswap AND

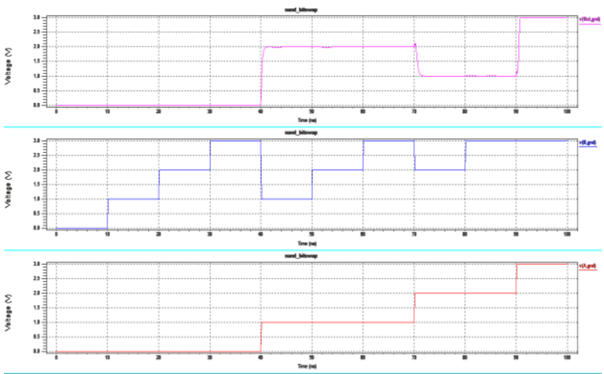

Fig10: Simulated input and output of bitswap AND

(d) Inward NOR

If inward inverter, followed by basic OR operator, we get the inward NOR. the symbolic representation of inward NOR is shown in fig 11, with its simulated input output characteristic in fig 12

Fig11: Symbol of Inward NOR

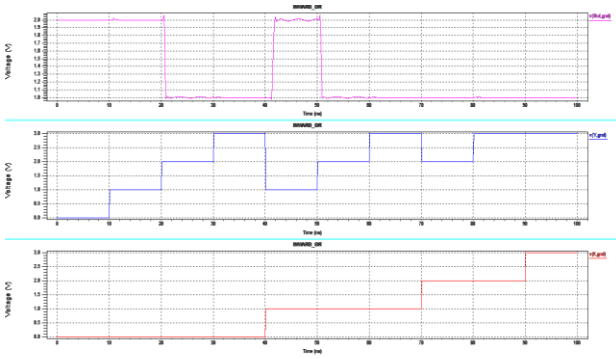

Fig12: Simulated input and output of inward NOR

(e) Outward NOR

Like basic inverter, if outward inverter , followed by basic OR operator, we get the outward NOr. the symbolic representation of Outward NOR is shown in fig 13, with its simulated input output characteristic in fig 14.

Fig13: Symbol of Outward NOR.

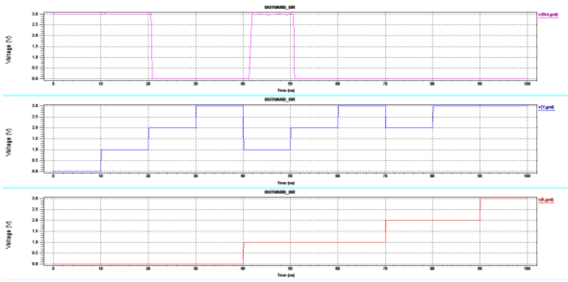

Fig14: Simulated input and output of Outward NOR

(f) Bitswap OR

If the binary bitswap, followed by basic OR operator, we get the bitswap OR. the symbolic representation of bitswap OR is shown in fig 15, with its simulated input output characteristic in fig 16.

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 2, Issue 4, April 2015

Fig15: Symbol of Bitswap OR

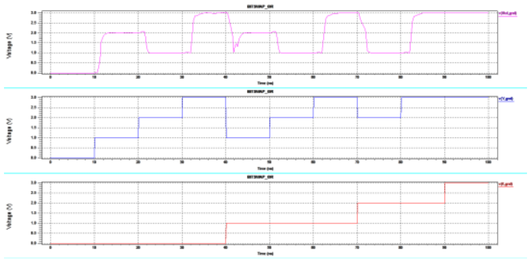
Fig16: Simulated input and output of Bitswap OR

(g) Special XOR operator

Like basic inverter, if inward inverter, Outward inverter and bitswap followed by basic XOR operator, we get the inward XNOR, Outward XNOR and bitswap XOR respectively. the symbolic representation of inward XNOR, outward XNOR and Bitswap XOR is shown in fig 17. Fig 18 and 19 shows the with simulated input output characteristic of XOR and XNOR

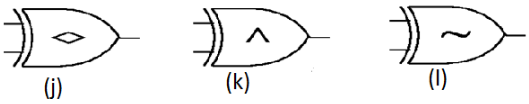

Fig17: Circuit Symbol of special XOR Operator (j)Inward XNOR. (k)Outward XNOR, (l) Bitswap XOR

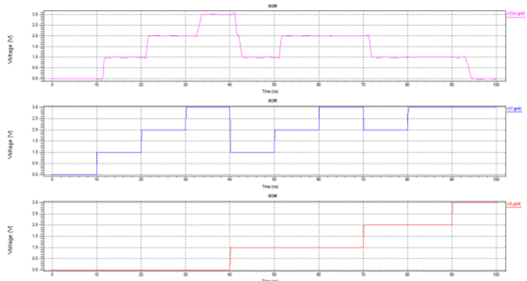
Fig18: Simulated input and output of XOR

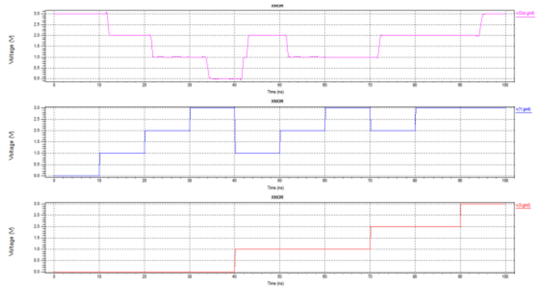
Fig19: Simulated input and output of XNOR

III. ELEMENTARY QUATERNARY ALGEBRA

The proposed quaternary logic scheme has its own algebra just like Boolean algebra in binary logic. Since basic operators are actually bitwise binary operators, the properties and axioms of Boolean algebra are applicable for basic quaternary operators. These include commutativity, associativity, distributivity, etc. Apart from the basic properties, the special operators have many other important properties of their own. These properties are indispensable in designing quaternary logic circuits. Some of these properties are as follows:

$$\overline{\overline{a}} = a, \quad \tilde{\tilde{a}} = a, \quad a'' \neq a, \quad \hat{\hat{a}} \neq a$$

$$\widetilde{a'} = \begin{cases} a+1 \,;\, a<2 \\ a.2 \,;\, a>1 \end{cases}$$

$$\tilde{a} + a = \begin{cases} 3 \,;\, a \neq 0 \\ 0 \,;\, a = 0 \end{cases}$$

$$\tilde{a}.a = \begin{cases} 3 \,;\, a = 3 \\ 0 \,;\, a \neq 3 \end{cases}$$

$$\tilde{a} \oplus a = \begin{cases} 3 \,;\, a \text{ asymmetric} \\ 0 \,;\, a \text{ symmetric} \end{cases}$$

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

$$a' + \hat{a} = \begin{cases} 1 \,;\, a>1 \\ 3 \,;\, a<2 \end{cases}$$

$$(a \widehat{+} b) = \hat{a}.\hat{b} \text{ and } (\widehat{a.b}) = \hat{a} + \hat{b}$$

$$(a \widetilde{+} b) = \tilde{a} + \tilde{b} \text{ and } (\widetilde{a.b}) = \tilde{a}.\tilde{b}$$

$$(a+b)' \neq a'.b', \ (a'.b') \neq a' + b'$$

$$(a+b)' \neq a' + b', \ (a'.b') \neq a'.b'$$

$$\hat{\overline{a}} = \overline{\hat{a}}, \ (\widehat{a'}) \neq (\hat{a})', \ (\overline{a})' = (\overline{a'})$$

$$\overline{\tilde{a}} = \tilde{\overline{a}}, \ \tilde{\hat{a}} \neq \hat{\tilde{a}}, \ (\widetilde{a'}) \neq (\tilde{a})'$$

$$\tilde{a} \oplus \tilde{b} = (a \widetilde{\oplus} b)$$

IV. CONCLUSION

In this paper we have proposed a new quaternary logic system. A set of operators are defined; some of which are closely related to Boolean operators, others are chosen for special purposes. Many properties of these operators are presented here and a functional quaternary algebra is developed. To check the functionality we have designed the special operators and their compound derivatives.

REFERENCES

[1]. Inaba, M; Tanno, K and Ishizuka,O, "Realization of NMAX and NMIN functions with multi-valued voltage comparators'', Multiple-Valued Logic, 2001. Proceedings, 31st IEEE International Symposium,pp.27 - 32, 22-24 May, 2001.

[2]. Mozammel H.A. Khan, "Quantum Realization of Quaternary Feynman and Toffoli Gates", Proceedings, ICECE 2006, 4th International Conference on Electrical and Computer Engineering, 19-21 December 2006, Dhaka, Bangladesh

[3]. http://www.ternarylogic.com

[4]. Hurst,S.L,"Multiple-Valued Logic; its Status and its Future",Computers, IEEE Transactions on Vol. C-33, Issue: 12,pp.1160-1179,Dec. 1984..

[5]. Jahangir, I.; Hasan, D. M. N.; Islam, S.; Siddique, N. A.; Hasan, M. M.; "Development of a Novel Quaternary Algebra with the Design of some Useful Logic Blocks", Proceedings, ICCIT 2009, 12th International Conference on Computers and Information Technology, 21-23 Dec. 2009

[6]. Jahangir, I.; Hasan, D. M. N.; Reza, M. S.; "Design of Some Quaternary Combinational Logic Blocks Using a New Logic System", Proceedings, TENCON 2009, IEEE Region 10 Conference, 2326 Nov. 2009

[7]. [7] Ricardo Cunha, Henri Boudinov and Luigi Carro, "Quaternary Look-up Tables Using Voltage-Mode CMOS Logic Design", Proceedings of the 37th International Symposium on Multiple-Valued Logic (ISMVL'07).

DOI 10.17148/IARJSET.2015.2418