Approaches to realize Canonical Form of Boolean Expression by using Reversible COG Gates

Shefali Mamataj
Assistant Professor
Department of ECE
Murshidabad College of Engineering & Technology

Biswajit Das
Assistant Professor
Department. of CSE
Murshidabad College of Engineering & Technology

ABSTRACT
Nowadays, reversible logic is one of the most important issue which has emerged as a promising technology having its applications in low power CMOS, quantum computing, nanotechnology and optical computing. Reversible logic circuits give less power dissipation as well as distinct output that is assigned for each distinct input. The classical set of gates such as the NAND, AND, NOR, OR, XOR and XNOR are not reversible. Reversible circuits are like the conventional logic circuits except that they are built from reversible gates. In reversible gates, there is a unique, one-to-one mapping between the inputs and outputs, which differ from the conventional logic. One of the most important factors for the acceptance of reversible logic lies in the fact that it can give a logic design methodology to design ultra-low power application beyond KTln2 limit for those emerging nanotechnologies in which the energy dissipated due to information destruction will be a significant factor of the overall heat dissipation. In this paper represents the approaches to realize the Canonical Form of Boolean Expression (CFOBE) by using reversible COG gates . For this, two methods are proposed and a comparison is also made between these two methods in terms of the number of reversible gates, constant input, garbage output and total logical calculation.

General Terms
Architecture, Logic Design, Reversible Logic.

Keywords
Reversible Logic, Reversible Gate, Boolean algebra, Canonical Boolean Expression, Garbage Output, Constant Inputs.

1. INTRODUCTION
The conventional logic gates dissipate a considerable amount of energy due to the lost of information bits during logic operations. According to Launder's principle, the loss of one bit of information dissipates kTln2 joules of energy where k is the Boltzmann's constant and T is the absolute temperature at which the operation is performed [1]. Later Bennett, in 1973, showed that in order to avoid kTln2 joules of energy dissipation in a circuit it must be built from reversible circuits [3]. Younis and Knight [12] showed that some reversible circuits can be made asymptotically energy-lossless if their delay is allowed to be arbitrarily large. According to Frank [2], reversible logic can recover a fraction of energy that can reach up to 100%. A reversible gate has the same number of inputs and outputs, inputs and outputs are related as one-to-one mapping [4]. A number of reversible gates have been designed till date. Some important basic reversible logic gates are Feynman gate [5] which is the only 2X2 reversible gate. T Toffoli gate [6], Fredkin gate [7], Peres gate [8] are 3X3 reversible gates all of which can be used to realize important combinational functions (See Figure 1).

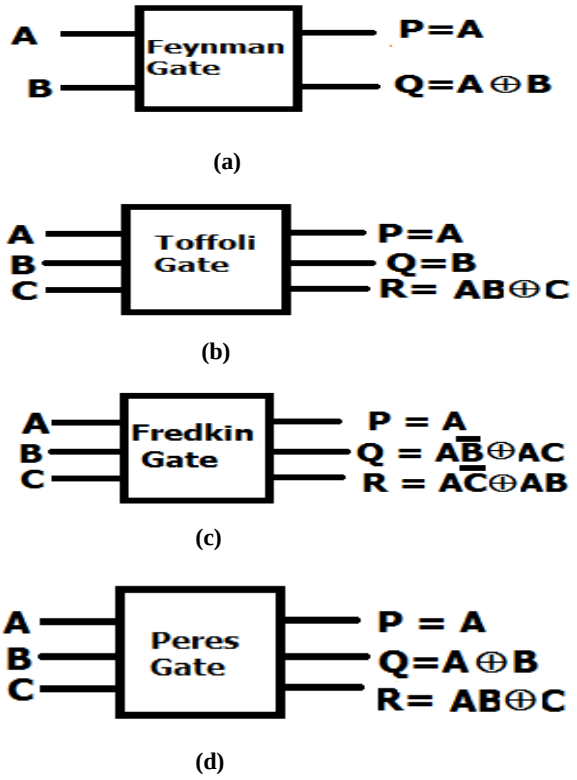

Fig 1: (a) Feynman gate, (b) Toffoli gate, (c) Fredkin gate and (d) Peres gate

A reversible logic circuit should have the following features [13]: Use minimum number of reversible gates, Use minimum number of garbage outputs, Use minimum constant inputs.

2. REVERSIBLE LOGIC
2.1 Definitions
Some of the basic Definitions [9] related to Reversible Logic are

2.1.1ReversibleLogicFunction
A reversible logic function is defined as a function for which each input vector maps to a unique output vector. From the given outputs, it is always possible to determine back its input in case of reversible function, because there is a one-to-one relationship between input and output states.

2.1.2ReversibleLogicGate
Reversible Gates are circuits in which number of outputs is

equal to the number of inputs and related to one to one mapping. For an N X N reversible logic gate the inputs are denoted by $I_1\ I_2\ I_3\ \ldots\ I_N$ and the outputs are denoted by $O_1O_2\ O_3 \ldots O_N$.

2.1.3Garbage

This also refers to the number of outputs which are not used in the synthesis of a given function. In certain cases these become mandatory to get reversibility. Garbage is the number of outputs added to make an n-input k-output function ((n; k) function) reversible.

2.1.4 Constant Inputs

These are the inputs that provide either a constant 0 or at constant 1 in order to generate a given logical expression using the reversible logic gates. The words constant inputs are used to denote the present value inputs that were added to an (n; k) function to make it reversible. The following simple formula shows the relation between the number of garbage outputs and constant inputs.

Input + constant input = output + garbage

2.1.5QuantumCost

The quantum cost (QC) of any reversible gate(circuit) is the number of 1×1 or 2×2 reversible gates and quantum logic gates which is required to realize the circuit.

2.1.6GateCount

This can be defined as the number of gates that are required to present a reversible logic circuit. .

2.1.7Flexibility

Flexibility can be defined in relation to the gate count, which signifies the ability of a reversible logic gate to realize more functions. Higher the flexibility of a gate needs lesser number of gates to implement a given function that means the gate count is lesser.

2.1.8TotalLogicalOperation

The total logical operation [10] is measured by counting the number of AND operations, number of EX-OR operations and number of OR operations. If α represents the number of EX-OR operations, β represents the number of AND operations and δ represents the number of NOT operations then the total logical operation T is given as sum of EX-OR, AND and NOT operations required for a specified circuit and can be expressed in terms of α, ß and δ.

2.2 COG Reversible Gates

A 3X3 reversible gate COG (Controlled Operation Gate) logic (See Figure 2) already had been proposed [11]. The truth table for the corresponding gate is shown in Table I also .The closer look at the truth table reveals that the input pattern corresponding to a specific output pattern can be uniquely determined and thereby maintaining that there is a one-to-one correspondence between the input vector and the output vector. In this gate the input vector is given by IV= (A, B, C) and the corresponding output vector is OV= (P, Q, R)

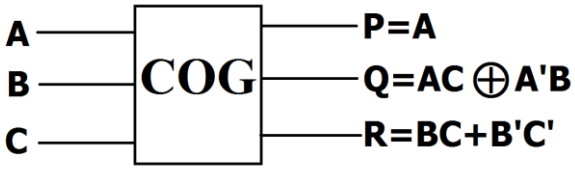

Fig 2: COG reversible gate

TABLE I. Truth table of COG gate

Inputs			Outputs		
A	**B**	**C**	**P**	**Q**	**R**
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	0	0	1
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	0	1
1	1	0	1	0	0
1	1	1	1	1	1

3. REALIZATION OF THE CLASSICAL OPERATION

Implementation of the conventional digital gates can be possible by using the COG reversible gate. Implementation of the AND, NOT, NAND, NOR, EXOR, EXNOR, OR and COPYING operations are possible (See Figure 3a to 3f). By making the inputs A=A, B=0 and C=B of COG gate AND, NOT & COPYING operation are found from the output lines(See figure 3a). In this way all the above specified operation can be get by setting the input values as per the requirement.

Fig 3a: .Implementation of AND, NOT & COPYING gate by COG Reversible gates

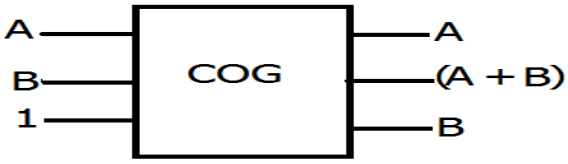

Fig 3b: Implementation of OR & COPYING gate by COG Reversible gates

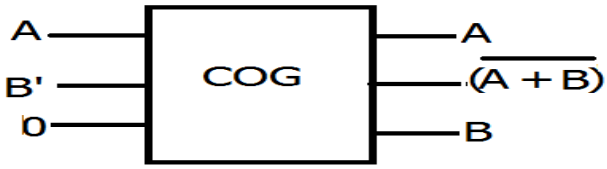

Fig 3c: Implementation of NOR & COPYING gate by COG Reversible gates

Fig 3d: Implementation of NAND, NOT & COPYING gate

by COG Reversible gates

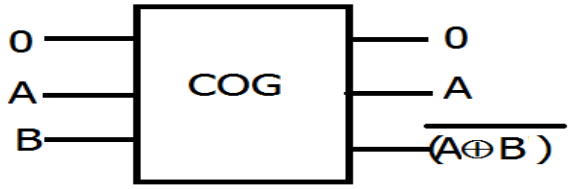

Fig 3e: Implementation of EXNOR & COPYING gate by COG Reversible gates

Fig 3f: Implementation of EXOR & COPYING gate by COG Reversible gates

4. CANONICAL FORM OF BOOLEAN EXPRESSIONS (CFOBE)

There are an infinite number of expressions for every Boolean function. A Boolean expression in canonical form is unique. We know there are two canonical forms SOP and POS. Both define a unique way to represent a Boolean function. So in Boolean algebra, any Boolean function can be put into the canonical disjunctive normal form (CDNF) or minterm canonical form and its dual canonical conjunctive normal form (CCNF) or maxterm canonical form. Minterms are called products because they are the logical AND of a set of variables, and maxterms are called sums because they are the logical OR of a set of variables shown in Table II for three variables. These concepts are dual because of their complementary-symmetry relationship as expressed by De Morgan's laws. There are 2^n minterms of n variables, since a variable in the minterm expression can be in either its direct or its complemented form—two choices per n variables. A given minterm n gives a true value (i.e., 1) for just one combination of the input variables. For example, minterm $5(m_5)$, $a\ b'\ c$, is true only when a and c both are true and b is false—the input arrangement where $a = 1$, $b = 0$, $c = 1$ results in 1.Now a three variable function for full adder sum can be expressed as Sum= (A⊕B⊕C) if the variables are A, B and C. So the equivalent Canonical SOP form of this function can be written as F(sum)=Σm(1,2,4,7) and its equivalent canonical POS form can be written as F(sum)= π M(0,3,5,6).These types of expressions are known as CFOBE .So from these CFOBE, the pattern can be easily known as per the requirements. For three variables X, Y and Z the minterms and maxterms combinations are shown in Table II.

TABLE II. Minterms and Maxterms for three variables

Variables			Minterms		Maxterms	
X	Y	Z	Term	Designation	Term	Designation
0	0	0	X'. Y'. Z'	m_0	X+Y+Z	M_0
0	0	1	X'. Y'. Z	m_1	X+Y+Z'	M_1
0	1	0	X'. Y. Z'	m_2	X+Y'+Z	M_2
0	1	1	X'. Y. Z	m_3	X+Y'+Z'	M_3
1	0	0	X. Y'. Z'	m_4	X'+X+Z	M_4
1	0	1	X. Y'. Z	m_5	X'+Y+Z'	M_5
1	1	0	X. Y. Z'	m_6	X'+Y'+Z	M_6
1	1	1	X. Y. Z	m_7	X'+Y'+Z'	M_7

5. DESIGN OF COMBINATIONAL CIRCUIT

Many combinational circuits can be designed by using the reversible COG gates; one of these circuits is multiplexer. Different types of multiplexers can be designed such as 2X1 MUX, 4X1 MUX, 8X1 MUX and 16X1 MUX (See Figure 4a, 4b, 4c and 4d).Generally multiplexer is used for multiplexing one channel out of many channels. .

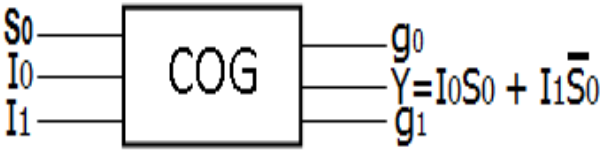

Fig 4a: 2 to 1 reversible Multiplexer

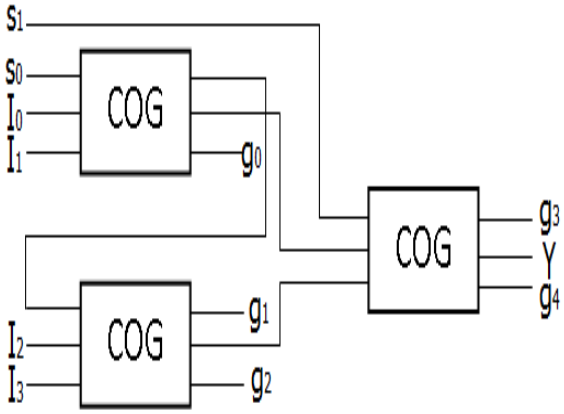

Fig 4b: 4 to 1 reversible Multiplexer

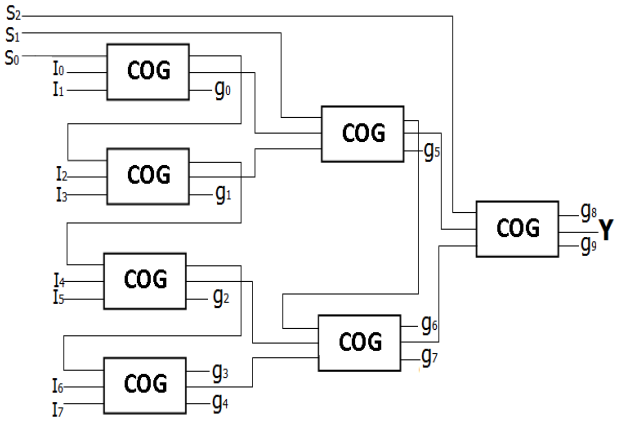

Fig 4c: 8 to 1 reversible multiplexer

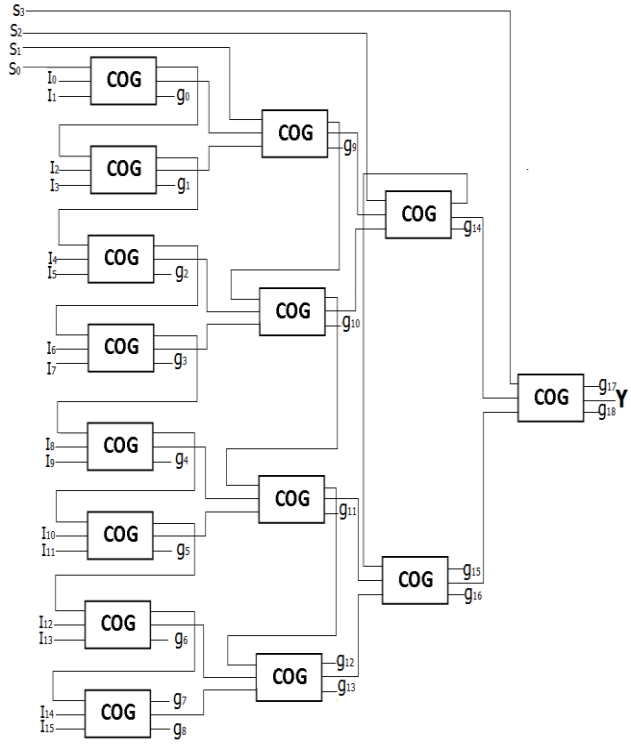

Fig 4d: 16 to 1 reversible Multiplexer

6. DESIGN APPROACHES TO REALIZE CFOBE

It is possible to realize the CFOBE by using reversible COG gates. Two methods of design approaches have been followed here for the realization of three variables CFOBE and four variables CFOBE.

6.1 Realization of 3 variables CFOBE

Realization of the three variables SOP form of CFOBE is done by using COG reversible gates in two ways. So a three variables SOP form of CFOBE, F(X, Y, Z) =Σm (2, 3, 5, 7) is assumed as an example to be implemented.

6.1.1 Method 1

In this method l, the SOP form of CFOBE, F(X, Y, Z) =Σm (2,3,5,7) has been realized by using 8X1 MUX of COG reversible gates (See Figure 5). Here the selection lines S_2, S_1 and S_0 are replaced by X, Y and Z respectively. Depending upon the value of X, Y and Z, the value of F(X,Y,Z) at the output line shown in Table IV. To realize the above said CFOBE seven COG gates and two constant inputs are required. The input lines I_2, I_3, I_5, I_7 of MUX are connected with Logic High (1) and I_0, I_1, I_4, I_6 input lines of MUX are connected with Low Logic(0). So whenever X=0,Y=1 & Z=0,the input line I_2 is selected. And as I_2 is connected with Logic High, the output value F(X, Y, Z) is 1(Logic High).Similarly the input lines I_3, I_5 & I_7 are selected for XYZ=011, XYZ=101 & XYZ=111 respectively and the output F(X,Y, Z) is bounded to reach at 1(Logic High) for these specified input patterns of X, Y & Z. But the input lines I_0, I_1, I_4 & I_6 are selected for XYZ=000, XYZ=001, XYZ=100 & XYZ=110 respectively. As because of the I_0, I_1, I_4, I_6 input lines of MUX are connected with Low Logic (0), the output value F(X, Y, Z) will be Low Logic (0). From the Table III the pattern of F(X, Y, Z) can be seen and it can be said that the output is High Logic (1) for the minterms m_2, m_3, m_5 & m_7.Hence CFOBE in SOP form for the output F(X, Y, Z) can be drawn from the Table III which is F(X, Y, Z) =Σm (2, 3, 5, 7) **.**Therefore the specified requirement can be made possible to fulfill in view of designing aspects by following the method1 **.** There are ten (10) garbage outputs in this method.

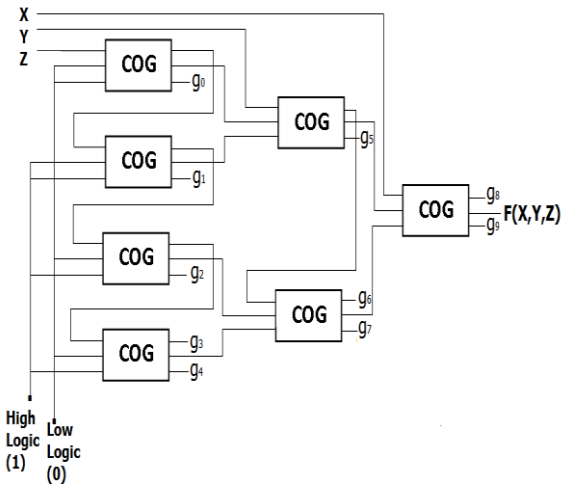

Fig 5: Realization of 3 variables expression by 8 to 1 reversible Multiplexer of COG gates

6.1.2 Method 2

In this method 2, the canonical SOP form of CFOBE, F(X,Y,Z)=Σm(2,3,5,7) has been realized here by using 4X1 MUX of COG gates (See figure 6).Here the selection lines S_1 and S_0 are replaced by Y and Z respectively. And another MSB variable X is used as input line. Depending upon the value of Y, Z and the input line X the value of F(X,Y,Z) can be achieved at the output line shown in Table IV. To realize the CFOBE in this method we need 3 COG gates and two constant inputs. The connection of the input lines of the MUX can be determined shown in Table III. The input line I_0 is connected with Low Logic (0), I_1 is connected with X, I_2 is connected with X' and I_3 is connected with High Logic(1). When X=0/1 & YZ=00, I_0 input line is selected .Now as I_0 is connected to Low Logic (0),the output F(X,Y,Z) is followed by Low Logic(0). Then X=0/1 & YZ=01choose the I_1 input line. Since I_1 is connected with X, the output value F(X,Y,Z) will depend on the value of X. F(X,Y,Z)=0 when X=0 and F(X,Y,Z)=1 when X=1. Again for X=0/1 & YZ=10 select the I_2 input lines. And as I_2 connected with the complement of X, the output F(X,Y,Z)=1 when X=0 and F(X,Y,Z)=0 when X=1. When X=0/1 & YZ=11, I_3 input line is selected .Now as I_3 is connected to High Logic (1),only the true value (logical 1)is transmitted to the output F(X,Y,Z) whatever may be the value

of X. So finally it can be noticed that the output value F(X,Y,Z) is 1(High Logic) for the minterms m_2,m_3,m_5 & m_7 .In total five garbage outputs are produced in this method.

TABLE III. Determination rule for MUX inputs

MSB variable	Input lines			
	I_0	I_1	I_2	I_3
X'	0	1	2	3
X	4	5	6	7
Values for input lines	0 (Low Logic)	X	X'	1 (High Logic)

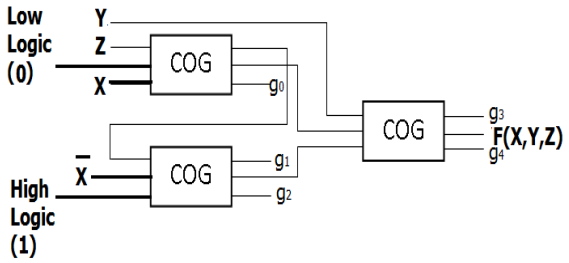

Fig 6: Realization of 3 variables expression by 4 to 1 reversible Multiplexer of COG gates

TABLE IV. Truth table for 3 variables expression F(X,Y,Z)=Σm(2,3,5,7)

Inputs			Outputs
X	Y	Z	F(X,Y,Z)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

From the truth table IV, it is observed that both the methods (method1 and method2) of 3 variables SOP form of CFOBE implementation are suitable for achieving the fruitful results. And the value of F(X,Y,Z) are high for the minterms m_2,m_3,m_5 and m_7 only and for rest of the minterms the value of F(X,Y,Z) are low.

6.2 Realization of 4 variables CFOBE

An example of four variables SOP form CFOBE,F(W,X,Y,Z) =Σm (0,1,3,4,8,9,15) is also considered for the realization .Again both the methods are followed here for this four variables SOP form CFOBE.

6.2.1 Method 1

In this method the four variables SOP form of CFOBE F(W,X,Y,Z) =Σm (0,1,3,4,8,9,15) has been realized by using 16X1 MUX of COG gates(See figure 7).Here the selection lines S_3, S_2, S_1 and S_0 are replaced by W,X,Y and Z respectively. Depending on the value of W, X, Y and Z the value of F(W,X,Y,Z) will be transmitted at the output line shown in Table V. To realize the CFOBE seven COG gates and two constant inputs are needed. The same rule like 3 variable implementation is followed here. The input lines I_0 , I_1 , I_3, I_4 I_8 ,I_9, I_{15} of MUX are connected with Logic High (1) and I_2 ,I_5, I_6, I_7 I_{10} ,I_{11}, I_{12}, I_{13} I_{14} input lines of MUX are connected with Low Logic(0). In total nineteen (19) garbage outputs are produced in this method of realization.

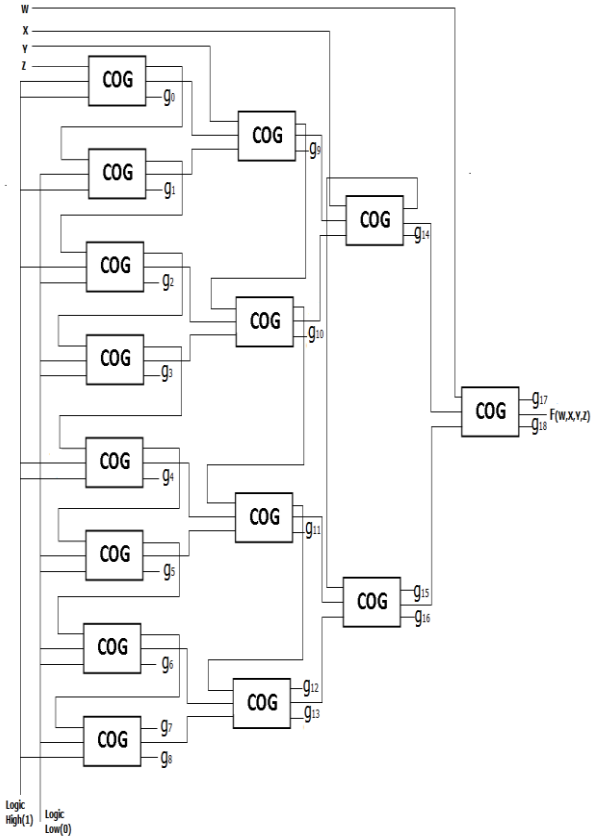

Fig 7: Realization of 4 variables expression by 16 to 1 reversible multiplexer

6.2.2 Method 2

In this method the SOP form of CFOBE, F(W,X,Y,Z) =Σm (0,1,3,4,8,9,15) has been realized by using 8X1 MUX (See figure 8) by following the same rule as followed by 3 variables CFOBE implementation in method 2. Here the selection lines S_2, S_1 and S_0 are replaced by the lower significant variables X, Y and Z respectively. And MSB variable W is used for setting the input lines. Depending on the value of X, Y, Z and the input line W the value of F (W,X,Y,Z) is achieved at the output line shown in Table V. To realize the above said CFOBE in this method seven COG gates and two constant inputs are required. The input lines I_0 I_1 are connected with High Logic(1),I_2 I_5 I_6 are connected with Low Logic(0) ,I_3 I_4 are connected with W' and I_7 is connected with W. Ten (10) garbage outputs are produced in this method.

In similar way n variables SOP form of CFOBE can be implemented with the help of COG reversible gates by following the above mentioned two methods.

For both the cases in method-1, all the variables of the given SOP form of CFOBE are used as the selection lines of the MUX and the input lines of MUX, specified by the minterms of the given CFOBE are connected with High Logic (1) and the rest of the input lines of the MUX are connected with Low Logic (0).In this method (2^nX1) MUX is required for the implementation where n is the number of variables required to express the SOP form of CFOBE..

But in method 2, all the variables except the MSB one of the given SOP form of CFOBE are used as the selection lines of the MUX and the input lines of MUX are connected as per the requirement specified by the minterms of the given SOP form of CFOBE. The input lines may be connected with High Logic (1), Low Logic (0), MSB variable or with the complemented form of MSB variable that have to be determined .Here (2^{n-1}X1) MUX is required for realization.

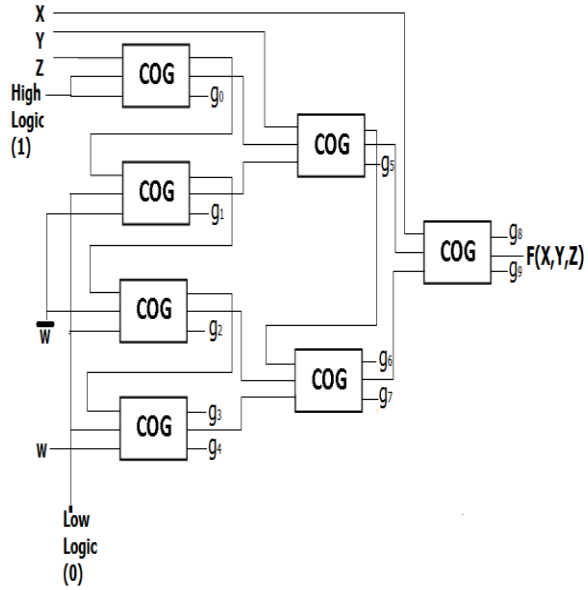

Fig 8: Realization of 4 variables expression by 8 to 1 reversible Multiplexer

TABLE V. Truth table for 4 variables expression F (W,X, Y, Z) =Σm (0,1,3,4,8, 9,15)

Inputs				Outputs
W	**X**	**Y**	**Z**	**F(X,Y,Z)**
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Again from the truth table V , both the cases (method 1 and method2) of 4 variables SOP form of CFOBE implementation are suitable for the fruitful results. And the value of F(W,X,Y,Z) are high for the minterms $m_0,m_1,m_3,m_4,m_8,m_9,m_{15}$ only and for rest of the minterms the value of F(W,X,Y,Z) are low.

7. COMPARISON RESULTS

A comparison is made between two proposed methods for both the three variables SOP form of CFOBE and the four variables SOP form of CFOBE shown in Table VI in terms of number of gates, garbage output, constant input and total logical calculation.

7. CONCLUSION

Two methods of implementation for both the three variables SOP form of CFOBE and the four variables SOP form of CFOBE have been followed and also compared shown in Table VI, in terms of number of gates, garbage output, constant input and total logical calculation. The general expression for the number of gates, garbage output, constant input and total logical calculation for n variables SOP form of CFOBE has also been derived for both the methods. Now because of their complementary-symmetry relationship between canonical SOP form and canonical POS form, the canonical POS form of Boolean expression by using these two methods can also be realized. It is concluded that the method 2 for canonical SOP form of CFOBE is more efficient with respect to method 1. It is also seen from the comparison Table V that in method2, less number of gates are required with respect to method 1. Also in terms of garbage output and total logical calculation method 2 is more suitable than method1. Implementation of the different type of applications like half adder circuit, full adder circuit, half subtractor circuit, full subtractor circuit, comparator and many more circuits can be possible by using these two methods. These two methods can also be efficiently used in the designing of ROM, PLA without minimizing the minterm functions. As a future work there is a vast application of these proposed design methods. Implementation of these design methods using quantum dot cellular automata and also the testing of its functionality by simulation for checking the correctness of these design methods can be treated as the possible future work.

ACKNOWLEDGMENTS

The authors wish to thank ECE Department and CSE Department of Murshidabad College of Engineering and Technology, Berhampore for supporting this work.

.

TABLE VI. Comparative results for method 1 and method 2.

Name of the Proposed circuits	Method 1				Method 2			
	No. of COG Gates	No. of Garbage output	No. of Constant Input	Total logical calculation	No. of COG Gates	No. of Garbage output	No. of Constant Input	Total logical calculation
3 variables CFOBE	7	10	2	14 α+14ß+14 δ	3	5	2	6 α+6ß+6 δ
4 variables CFOBE	15	19	2	30 α+30ß+30 δ	7	10	2	14 α+14ß+14 δ
n variables CFOBE	(2^n-1)	(2^n+n-1)	2	(2^n-1) (2 α+2ß+2 δ)	$(2^{n-1}-1)$	$(2^{n-1}+n-1)$	2	$(2^{n-1}-1)$ (2α+2ß+2 δ)

8. REFERENCES

[1] Rolf Launder, Irreversibility and Heat Generation in the Computing Process",IBM Journal of Research and Development, vol. 5, pp. 183-191, 1961.

[2] M. P. Frank. Introduction to reversible computing: motivation, progress, and challenges. In Proceedings of the 2nd Conference on Computing Frontiers, pages 385-390, Ischia, Italy, 4-6 May2005.

[3] Charles.H.Bennett, Logical Reversibility of computation, IBM Journal of Research and Development, vol. 17, no. 6, pp. 525-532, 1973.

[4] Md Selim Al Mamun, Indrani Mandal and Md Hasanuzzaman. "Design of Universal Shift Register Using Reversible Logic." (2012).

[5] Richard P.Feynman, "Quantum mechanical computers," Foundations of Physics, vol. 16, no. 6, pp 507-531, 1986.

[6] Tommaso Toffoli, "Reversible Computing," Automata, Languages and Programming, 7th Colloquium of Lecture Notes in Computer Science, vol. 85, pp. 632-644, 1980.

[7] Edward Fredkin and Tommaso Toffoli, "Conservative Logic," International Journal of Theoretical Physics, vol. 21, pp. 219-253, 1982

[8] A. Peres, "Reversible Logic and Quantum Computers," Physical Review A, vol. 32, pp. 3266-3276, 1985

[9] Rakshith Saligram and Rakshith T R "Novel Code Converter employing Reversible Logic" Intl. Journal of Computer Applications, Vol 52, No. 18, Aug 2012.

[10] Md. Saiful Islam et.al" Synthesis of fault tolerant Reversible logic" IEEE 2009.

[11] Shefali Mamataj, Biswajit Das, Anurima Rahaman "An Ease implementation of 4-bit Arithmetic Circuit for 8 Operation by using a new reversible COG gate" International Journal of Advanced Research in Electrical , Electronics and Instrumentation Engineering Vol. 3, Issue 1, January 2014.

[12] S.Younis and T. Knight, "Asymptotically Zero Energy Split-Level Charge Recovery Logic, "Workshop on Low Power Design, June1994

[13] Perkowski, M. and P. Kerntopf, Reversible Logic. Invited tutorial" Proc. EURO-MICRO, Warsaw, Poland ,Sept 2001.