Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

Design of Reversible Fault Tolerant Programmable Logic Arrays with Vector Orientation

Rubaia Rahman, Lafifa Jamal, Hafiz Md. Hasan Babu
Department of Computer Science & Engineering, University of Dhaka, Dhaka-1000, Bangladesh

ABSTRACT

In recent years, reversible logic has emerged as a promising computing paradigm having application in low power CMOS, quantum computing and error detecting. Reversible computing dissipates zero energy in terms of information loss and also it can detect error of circuit by keeping unique input-output mapping. In this paper, we have proposed a regular structure of Reversible Fault Tolerant Programmable Logic Arrays (RFTPLAs) and corresponding algorithms for construction of RFTPLAs. Proposed algorithms can realize ESOP (Exclusive Sum-of-Products) operations in terms of multi-output functions by using minimum numbers of gates, garbage bits and quantum costs. In our design the novel properties of FRG (Fredkin Gate) and F2G (Feynman Double Gate) gates are used for the realization of proposed designs. Finally, we compare the proposed design with existing RPLA.

Keywords: *Reversible Computing, RPLAs, Fault tolerance, Fredkin Gate, Feynman Double Gate, AND-EX-OR*

I. INTRODUCTION

Conventional circuit lost information because the input vector cannot be uniquely recovered from the output vector. Bennett and Landauer [1] [2] proved that information loss causes power loss. Landauer's[2] principle states that logic computations that are not reversible necessarily generate KT*log2 Joules energy for every bit of lost information, where K is the Boltzmann's constant and T is the absolute temperature at which the computation is performed. A reversible circuit does not loss information.

Programmable Logic Devices such as PLA, PAL etc use the array of conventional gates which are not reversible except NOT. Fleisher and Maissel[3] introduced array logic based on AND, OR and NOT synthesis to implement SOP or POS. But reversible logic prefers EX-OR operation as well as ESOP synthesis. ESOP synthesis gives out better result than SOP realization where many useful methods are proposed for minimizing multi-output Boolean functions into ESOP form [4],[5]. A regular structure of reversible wave cascade of ESOP synthesis is proposed in [6]. Realization of reversible functions and garbage minimization technique is proposed in [7]. The generalized structure of Reversible Programmable Logic Array was first proposed in [7] which are based on ESOP realization of multi-output functions.

Various design of PLA has been proposed previously but none of them has fault detection capability. Our work combines several ideas to present the construction of Reversible Fault Tolerant Programmable Logic Arrays for multi-output functions by introducing new structure based on Feynman Double gates and Fredkin gates; Heuristic algorithm for proposed RPLA; Ordering of functions as well as ordering of Products; Optimization of EX-OR plane; AND plane minimization in term of Products; Comparative study with existing design. Finally, we end the paper with concluding remarks.

II. BASIC DEFINITIONS AND PROPERTIES

In this section, we have discussed the Reversible logic, architecture of Programmable Logic Array (PLA) and Quantum realization of reversible circuit.

A. Reversible Logic

Reversible computing is the only method to recover bit loss by using unique mapping between input and output vectors.

Definition 1: Frequently used Reversible logics are composed into gate level called **Reversible Gate** in which the number of inputs is equal to the number of outputs having a unique mapping between input and output vectors [8].

Let, the input vector, I_v $\{I_v = I_1, I_2, I_3, ..., I_n\}$ and the output vector, O_v $\{O_v = O_1, O_2, O_3, ..., O_n\}$ of any Reversible Gate then according to the above definition the relation between two vectors is, $I_v \leftrightarrow O_v$. Fig. 1 shows the perspective comparison between conventional (irreversible) and reversible circuits for the EX-OR operation.

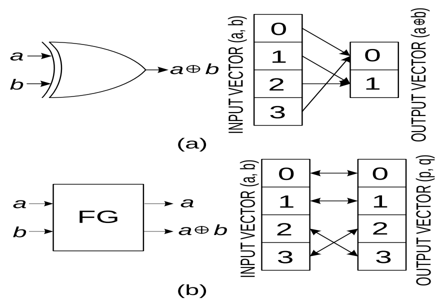

Fig. 1. (a) Conventional circuit for EX-OR operation and corresponding input and output mapping and (b) Reversible implementation of EX-OR operation and Input Output unique mapping

Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

Definition 2: The input vector, I_v and output vector, O_v for 2x2 **Feynman Gate (FG)** [9] is defined as follows: $I_v = (a, b)$ and $O_v = (a, a \oplus b)$. Fig. 1(b) shows the block diagram of Feynman gate and the mapping between input-output vectors which is unique.

Definition 3: The unused outputs of any reversible gate or circuit is called **Garbage Output** which will never been used in future rather than to check reversibility [8].

Example 1: One Feynman Gate can be used to implement reversible EX-OR operation which generates a dummy output along with its principle output signal to preserve reversibility. The garbage output is denoted by ***p*** in Fig. 1(b).

Definition 4: The input vector, I_v and output vector, O_v of 3x3 **Fredkin Gate (FRG)** [10] can be defined as follows: $I_v = (a, b, c)$ and $O_v = (a, a'b \oplus ab, a'c \oplus ab)$. The pictorial representation of FRG is shown in Fig. 2(a).

Definition 5: The input vector, I_v and output vector, O_v of 3x3 **Toffoli Gate (TG)** [11] (shown in Fig. 2(b)) can be defined as follows: $I_v = (a, b, c)$ and $O_v = (a, b, ab \oplus c)$.

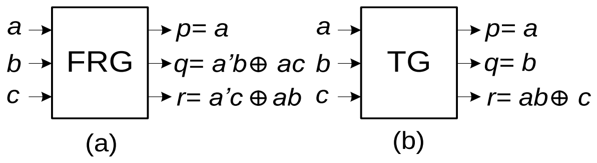

Fig. 2: (a) Fredkin Gate and (b) Toffoli Gate

Definition 6: The input vector, I_v and output vector, O_v of 3x3 **Feynman Double Gate (F2G)** [14] can be defined as follows: $I_v = (a, b, c)$ and $O_v = (a, a \oplus b, a \oplus c)$.

B. Fault Tolerant Method

Reversibility recovers bit loss but is not able to detect bit error in circuit. Fault Tolerant reversible circuit is capable to prevent error at outputs.

Definition 7: Fault Tolerant (FT) gate, also called Conservative Reversible Gate [14] which means the Hamming weight of its input and output are equal.

For example, the Fault Tolerance property of Feynman Double gate can be verified from Fig. 3 where square and circle represents ODD and EVEN parities respectively and the equivalent decimal values of input-output vectors are represented as corresponding decimal number (0-7).

F2G, FRG and NFT are 3x3 dimensional and MIG is 4x4 dimensional fault tolerant gates having unique mapping between input and output vectors. The input and corresponding output parities of fault tolerant gates are same [8]. Fault tolerant gate is also called parity preserving gate.

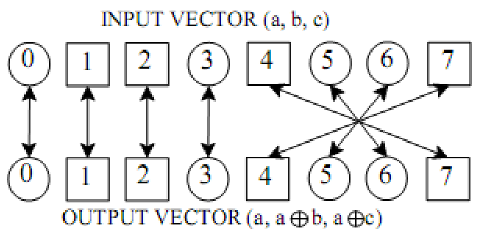

Fig. 3. Input –Output of Feynman Double Gate

C. Quantam Realization of Reversible Circuit

Quantum Computation is gaining popularity as some exponentially hard problem can be solved in polynomial time [12] and reversibility can be used to construct quantum circuits [8]. Quantum realization is another fact to judge the efficiency of reversible circuit. Quantum computation uses matrix multiplication rather than conventional Boolean operations and the information measurement is realized using qubits rather than bits. The matrix operations over qubits are simply specifies by using quantum primitives are shown in Fig. 4. The value of qubits is the probability factor of 0 or 1 which are represented as |0> or |1> where

$$|0\rangle = \alpha|0\rangle + \beta|1\rangle \; and \; |1\rangle = \alpha|1\rangle + \beta|0\rangle$$

Definition 8: The **Quantum Cost (QC)** of any reversible circuit is the total number of 2x2 quantum primitives which are used to form equivalent quantum circuit.

Example 2: The quantum cost of reversible Feynman Gate (shown in Fig. 4(b)) is one because single 2x2 quantum EX-OR is able to realize Feynman gate. Fig. 5 shows the quantum representation of Fredkin Gate where square box represent symmetric gate pattern which has a cost of one [13]. From Fig 5. we can see that the quantum cost of Fredkin Gate is 5. Similarly, we can calculate the quantum cost of Feynman Double Gate which is 5.

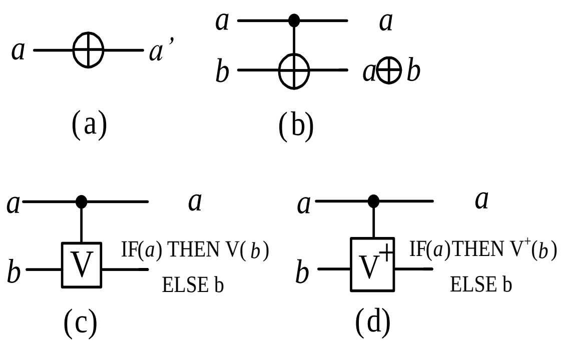

Fig. 4. Quantum primitives are used for realization of Reversible circuits: (a) NOT (b) EX-OR (c) Square Root of NOT (SRN) and (d) Hermitian of SRN

Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

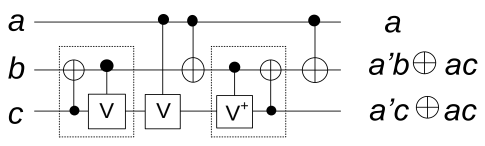

Fig. 5. Quantum Realization of Fredkin Gate, cost is 5.

III. PROGRAMMABLE LOGIC ARRAY

Programmable Logic Array allows Sum-Of-Products (SOP) for implementing Boolean functions. The typical implementation consists of input buffers, the programmable AND-matrix followed by the OR-matrix, and output buffers.

Definition 9: Programmable Logic Array (PLA) consists of two planes, the first one is AND plane and the second one is OR plane known as AND-OR PLA. When the second plane works as EX-OR, then it is called AND-EXOR PLA [7].

Definition 10: AND plane generates product by combining two or more literals. In our proposed design, AND plane generates product which uses two types of gates: Feynman Double Gate which is used to copy or recover fan-out problem and Fredkin Gate which is used for AND purpose.

Definition 11: EX-OR plane uses Products which come from AND plane and produces final output functions by EX-ORing those products zero or more times. Here we have designed the EX-OR plane by using only Feynman Double Gate.

Example 3: Fig. 6 shows the design of AND-EXOR PLA consists of AND plane followed by EX-OR plane. AND plane takes inputs whereas EX-OR plane generates output functions and the products are passed from AND plane to EX-OR plane by using the intermediate lines.

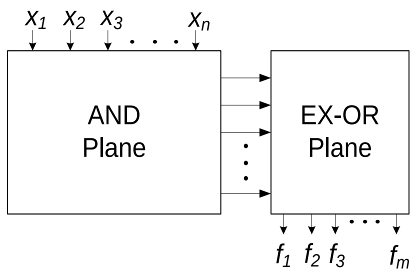

Fig. 6. AND-EXOR Programmable Logic Arrays

IV. REVERSIBLE PROGRAMMABLE LOGIC ARRAY

In this section, first we have discussed about the existing design of RPLA [7] and its limitations then we proposed a regular architecture of proposed reversible PLA based on order of output functions and input variables.

A. Existing Design of RPLA's

The design of Reversible PLA proposed in [7] uses Feynman and Toffoli gates to realize Reversible PLA for multi-output ESOP operations. Existing design has used Toffoli gate for AND operation and Feynman gate for EX-OR operation in AND plane and EX-OR plane respectively. Let, 1 is a multi-output ESOP function, where $F = \{f_1, f_2, f_3, f_4, f_5\}$ as follows:

$$\left.\begin{aligned} f_1 &= ac \oplus a'b'c \\ f_2 &= ab' \oplus ab'c \\ f_3 &= ab' \oplus ab'c \oplus bc' \\ f_4 &= ac \\ f_5 &= ab' \oplus ac \oplus bc' \end{aligned}\right\} \cdots\cdots\cdots (I)$$

The realization of multiple-output function F (Equation (I)) based on existing algorithm is shown in Fig. 7. But the existing design has following limitations:

1. Treated primary input as garbage when it becomes as an output
2. Used conventional architecture (Complement and non-complement lines for copying input variables)
3. The design is not fault tolerant
4. Reduced the cost of EX-OR plane but no algorithm has proposed for minimization of AND plane

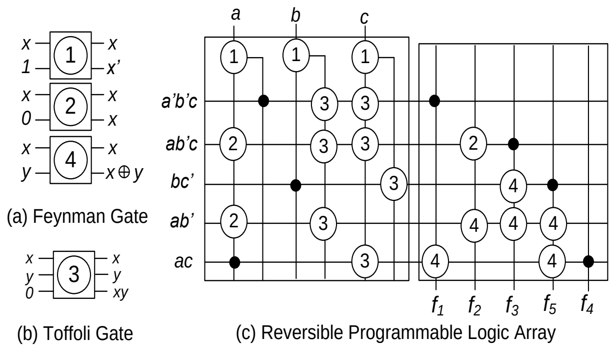

Fig. 7. Realization of multi-output function *F* by using existing design

The existing design used Toffoli gate for AND operation. So, there was no way to generate products having complemented form of one or more literals as output without complementing the input variable(s). For this reason, Toffoli gate produces huge number of unused outputs which enforced the design towards to such type of definition of garbage and there is no scope to minimize AND plane.

Rest of the part of this section, we have discussed the proposed idea of the construction of RPLA and in the next section we have shown about the minimization technique by using ordering of input variables.

B. Proposed Design

Proposed design is based on the orientation of input output vectors of Feynman Double (F2G) and Fredkin (FRG) gates. The input and output vectors of any reversible circuit or gate contains a unique mapping and the orientation of the input and the output vector also preserves the uniqueness. For example, Fig-8(a) shows two orientations (F2G-1 and F2G-2) of Feynman Double gate and Fig 8(b) shows two orientations (FRG-3 and FRG-4) of Fredkin gate. Input *a* and *c* (output *p* and *r*) of Feynman Double gate swap their position to make a different pattern of Feynman gate (shown in Fig. 8(a)). On the other hand, output *q* and *r* of Fredkin gate swap their position and create another pattern of Fredkin (shown in Fig. 8(b)).

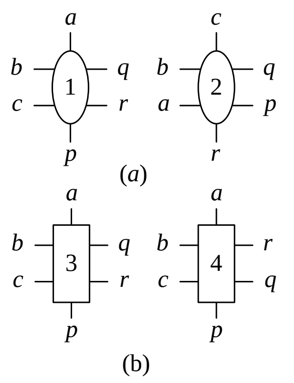

Fig. 8. Four different orientations of (a) Feynman Double gate and (b) Fredkin gate

The order of products will be generated after the optimization of EX-OR plane. In this subsection, we have proposed two algorithms on the minimizations and construction of EX-OR plane followed by realization of AND plane for generalizing proposed design.

Definition 12: Frequency of a Product is the total number of output functions which will use this product.

Example 9: Table I shows the Frequency of all Products of the multi-output function the *F* {f_1, f_2, f_3, f_4, f_5}. For instance, the frequency of *ac* is the highest i.e., 3 and a'b'c' is the lowest i.e. 1.

Table I: Frequency Matrix based on multi-output function, F

Products	**Function**					***Frequency of Product***
	f_1	f_2	f_3	f_4	f_5	
a'b		X	X		X	3
a'bc		X	X			2
ac	X			X	X	3
bc'			X		X	2
a'b'c	X					1

Definition 13: The cross point in RPLA, in which no gate is used, is termed as **DOT** [7].

Example 4: In Fig 9. Each of function f_1, f_2, f_3 and f_4 uses DOT at top most horizontal line.

Definition 14: Number of Product of a function is the total number of products that are used to realize the function in ESOP form.

Example 4: The number of product of function f_1 is 2 because two products are required to realize function f_1 in ESOP form. Table II shows the number of product of each function individually.

Table II: Calculation of Number of Product of Functions

Functions	f_1	f_2	f_3	f_4	f_5
Number of product	2	2	3	1	3

In the proposed design, the order of output functions is related to the number of product. Functions are generated in ascending order based on this criterion, the realization algorithm of EX-OR plane is defined as follows:

Algorithm 1: Construction_Of_EX-OR_Plane

```
1.  TDOT:= 0 [TDOT= Total number of DOT]
2.  Sort output Functions according to their number of product
3.  REPEAT Step 4 for each output function f_i of F
4.  IF N_O_P of f_i is one THEN
5.      IF Product P_j exist THEN use F2G-2
6.      ELSE Generate P_j in above line and use DOT
            TDOT:= TDOT+ 1
7.      END IF
8.  ELSE
9.      IF all Product(s) P_j exist THEN use F2G-2 for all
10.     ELSE Generate Product(s) (P_j) on upper line(s) according to Freq. of Product and use DOT for top most and F2G-2 for exist and newly generated Products
            TDOT := TDOT+ 1
11.     END IF
12. END IF
End
```

Theorem 1: The minimum number of Feynman Double gates to realize EX-OR plane is (*n* + *m* - *TDOT*) where,

m = Number of output functions

n = Number of EX-OR operations in *m*

TDOT = Number of cross points (DOT)

Proof: When there are *TDOT* cross points for *m* functions the number of additional Feynman Double gates in EX-OR plane of RPLA is *m-TDOT*. As there are *n* EX-OR operations by *n*-Feynman Double gates. The total number of Feynman Double gates in the EX-OR plane of RPLA is $f = n + m - TDOT$.

□

Example: For multi-output function *F* of Subsection 3.1, number of outputs (*m*) is 5 and number of EX-OR operations are 6. Consider Fig. 8, number of *TDOT* is 4. So the number of Feynman Double gates, $n + m - TDOT = 6 + 5 - 4 = 7$.

By using proposed algorithm the realization of EX-OR plane for function *F* is shown in Fig. 9.

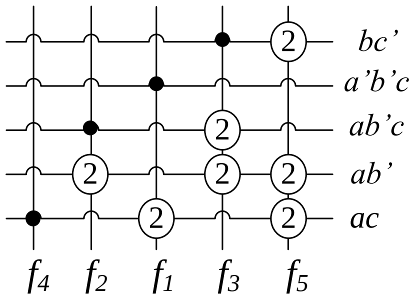

Fig. 9: Realization of multi-output function (*F*) based on proposed Algorithm 1

Theorem 2: Let, *P* be the number of Products, *TDOT* be the number of cross-points and *x* be the number of horizontal series in the EX-OR plane of RPLA. The minimum number of garbages to realize EX-OR plane of RPLA is $(P - TDOT + x)$.

Proof: As there are *TDOT* cross points and *P* Products for *m* output functions, the total number of garbage outputs in the EX-OR plane of RPLA is $P\text{-}TDOT + x$, where x be the number of horizontal series in the EX-OR plane of RPLA.

Example: Consider Fig. 9, for multi-output function *F* of Subsection 3.1. In Fig. 9, number of Products (*P*) is 5 and number of cross points (*TDOT*) is 4. Horizontal series of F2G gates propagate a garbage bits until the end and the number of horizontal series is 4. So the number of garbage outputs, $P\text{-}TDOT + x = 5 - 1 + 4 = 8$.

The realization of EX-OR plane generates the order of Products shown in Fig. 9. AND plane will be constructed according to order of Products which is designed by EX-OR Plane. Following algorithm (Algorithm 2) describes the construction of AND plane by using Fredkin and Feynman Double gates.

Algorithm 2: Construction_Of_AND_Plane

```
1.  TDOT:= 0 [TDOT= Total number of DOT]
2.  REPEAT 3 for each ordered product ($P_i$)
3.  IF $l_j$ is the first literal THEN
4.      IF $l_j$ is in complemented form THEN apply F2G-1
5.      ELSE
6.          IF $l_j$ is further used THEN apply F2G-1
7.          ELSE use DOT and
8.              TDOT := TDOT + 1
9.          END IF
10.     END IF
11. ELSE use FRG-4 and FRG-3 based on ANDing patter of any product ($P_i$)
12. END IF
End
```

In AND plane, the Feynman Double gates are used to recover fan-out problem and the Fredkin gates are used for AND operations. The generation of complementary form of input literals in AND plane is unnecessary in AND plane because Fredkin gate is able to generate products with or without complement form of literal which Toffoli gate can't. By using Algorithms 1 and 2, the realization of proposed Reversible PLA is shown in Fig. 10.

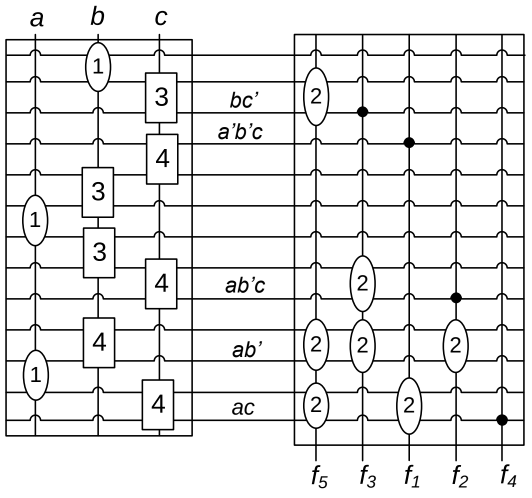

Fig. 10. Realization of multi-output function *F* based on proposed Algorithms 1 and 2

In this section, EX-OR plane realization requires minimum number of gates and garbage outputs based on Algorithm 1. On the other hand, Algorithm 2 describes the construction of AND plane. Table III shows the comparison between proposed and existing [7] designs where the cost factors are comparatively less than existing design.

Table III. Comparison between proposed and existing [7] designs

Reversible PLA	Total Gates	Garbage	Quantum cost	Fault Tolerant
Proposed design	17	19	55	Yes

Existing Design [7]	19	11	47	No

Table III shows the comparison between the proposed and existing [7] designs where we can see that the proposed design needs less number of gates than the existing design. Here we can see that here the garbage bit and quantum cost increase for the proposed design. But the main achievement of the proposed design is that it is **Fault Tolerant** whereas the existing design is not fault tolerant.

V. CONCLUSIONS

In this paper, we proposed a regular structure of Reversible Fault Tolerant Programmable Logic Arrays (RFTPLAs) by using the different vector orientations of input and output of Feynman Double gate and Fredkin gate and also presented the minimization techniques for both AND and EX-OR planes. We used the garbage outputs as operational outputs that reduced the number of AND operations in RFTPLAs. Finally, we figured the performance of proposed design over existing design by using pictorial representation. RPLAs are useful in embedded circuits and other technologies for power consumption and fault tolerant [3], [11].

REFERENCES

[1] C. H. Bennett, "Logical reversibility of computation", *IBM J. Research and Development*, 17, pp. 525-532, 1973.

[2] R. Landauer, "Irreversibility and heat generation in the computing process", *IBM J. Research and Development*, vol- 5, pp. 183-191, 1961.

[3] H. Fleisher and L. I. Maissel, "An introduction to array logic", *IBM Journal of Research and Development*, vol- 19, 1975.

[4] T. Sasa, "Logic Synthesis and optimization", *Kluwer Academic Publisher*, 1993.

[5] T. Sasao, "Exmin2: A simpli_cation algorithm for exclusive-or-sum-of -products expressions for multiple-valued input two-valued output functions", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol- 12, no- 5, pp. 621-632, May 1993.

[6] M. Perkowski, A. B. P. Kerntof, M. Chrzanowska-Jeske, A. Mishchenko and et al, "Regularity and symmetry as a base of efficient realization of reversible logic circuits", *In International Workshop on Logic Synthesis*, pp. 245-252, June, 2001.

[7] A. R. Chowdhury, R. Nazmul and H. M. H. Babu, "A new approach to synthesize multiple-output functions using reversible programmable logic array". *In 19th International Conference on VLSI Design held jointly with 5th International Conference on Embedded Systems Design (VLSID'06)*, pp. 311-316, 2006.

[8] A. K. Biswas, M. M. Hasan, A. R. Chowdhury and H. M. H. Babu, "Efficient approaches for designing reversible binary coded decimal adders", *Microelectronics Jounrnal*, vol- 39, December (2008)

[9] R. Feynman, "Quantum mechanical computers", *Opt. News*, vol- 11, no- 2, pp. 11-20, 1985.

[10] E. Fredkin and T. Toffoli, "Conservative logic", *Int'l Journal Theoretical Physics*, vol- 21, pp. 219-253, 1982.

[11] T. Toffoli, "Reversible computing.", *MIT Lab for Comp. Sci.*, vol- 85, pp. 632-644, 1980.

[12] V. V. Shende, A. Prasad, I. Markov and J. Hayes, "Reasoning about naming systems". *IEEE Trans. CAD*, vol- 22, no- 6, pp. 723-729, 2003.

[13] H. N. N. William, S. Xiaoyu, Y. Guowu, Y. Jin and M. Perkowski, "Quantum logic synthesis by symbolic reachability analysis", *In 41st ACM/IEEE. Design Automation Conference*, pp. 838-841, May, 2004.

[14] B. Parhami, "Fault tolerant reversible circuits," *in In Proc. of 40th Asimolar Conference Signals, Systems and Computers*. Pacific Grove, CA, pp. 1726–1729, 2006.