An Efficient Approach for Designing and Minimizing Reversible Programmable Logic Arrays

Sajib Kumar Mitra
Samsung Bangladesh R&D Center Ltd.
57&57/A Gulshan Avenue (I)
Dhaka-1212, Bangladesh
sajib.mitra@samsung.com

Lafifa Jamal
Dept. of Computer Sci. and Engineering
Faculty of Engg. and Tech.
University of Dhaka
Dhaka-1000, Bangladesh
lafifa@yahoo.com

Mineo Kaneko
School of Information Science
Japan Advanced Institute of Sci. and Tech.
1-1 Asahidai, Ishikawa
932-1292 Japan
mkaneko@jaist.ac.jp

Hafiz Md. Hasan Babu*
Dept. of Computer Sci. and Engineering
Faculty of Engg. and Tech.
University of Dhaka
Dhaka-1000, Bangladesh
hafizbabu@hotmail.com

ABSTRACT

Reversible computing dissipates zero energy in terms of information loss at input and also it can detect error of circuit by keeping unique input-output mapping. In this paper, we have proposed a cost effective design of Reversible Programmable Logic Arrays (RPLAs) which is able to realize multi-output ESOP (Exclusive-OR Sum-Of-Product) functions by using a cost effective 3×3 reversible gate, called MG (MUX Gate). Also a new algorithm has been proposed for the calculation of critical path delay of reversible PLAs. The minimization processes consist of algorithms for ordering of output functions followed by the ordering of products. Five lower bounds on the numbers of gates, garbages and quantum costs of reversible PLAs are also proposed. Finally, we have compared the efficiency of proposed design with the existing one by providing benchmark functions analysis. The experimental results show that the proposed design outperforms the existing one in terms of numbers of gates, garbages, quantum costs and delay.

Categories and Subject Descriptors

B.6.1 [**Logic Design**]: Design Styles—*combinational logic, logic arrays*

General Terms

Design, Algorithm, Performance

*Corresponding Author

GLSVLSIŠ12, May 3–4, 2012, Salt Lake City, Utah, USA.

Keywords

Reversible Logic, Programmable Logic Arrays, MUX Gate

1. INTRODUCTION

Programmable Logic Devices such as PLA, PAL or GAL etc use the array of conventional gates which are not reversible except NOT. Such type of irreversible circuit dissipates $kT{*}log2$ joules of heat energy to reload information per bit [1, 2], where k is the Boltzmann's constant and T is the absolute area temperature. The performance is not so pleasant rather reversible computing drives multiple operations in a single cycle [3]. Reversible circuits are of particular interest in low power CMOS design [4], optical computing [5], quantum computing [6] and nanotechnology [7].

Array Logic was introduced by Fleisher and Maissel [8] based on AND, OR and NOT synthesis to implement SOP or POS whereas Reversible Logic prefers EX-OR operation as well as Exclusive Sun-Of-Product (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 [9], [10]. A regular structure of reversible wave cascade of ESOP synthesis is proposed in [11]. Cascade realization of reversible functions and garbage minimization technique is proposed in [12]. The generalized structure of Reversible PLA was first proposed in [13] based on ESOP realization of multi-output functions. Finally, this paper has proposed a new approach of designing Reversible Programmable Logic Arrays as well as compared the proposed design with existing [13] one.

Our work significantly advances the design of cost effective Reversible Programmable Logic Arrays by combining the overview of the design of 3×3 MUX gate [14]; introducing the architecture of RPLAs by using Feynman and MUX gates; novel approach for calculating delay of RPLAs; comparison with existing design and performance analysis by using different benchmark functions.

2. BASIC DEFINITIONS AND PROPERTIES

In this section, we have discussed the basic phenomena of reversible logic, quantum realization and cost calculation of reversible circuits including the architecture of Programmable Logic Arrays (PLAs).

2.1 Reversible Logic

Reversible Computing is the only one method to recover bit loss by using unique mapping between input and output vectors. Frequently used logical operations are composed into gate level called ***Reversible Gate*** where the number of inputs is equal to the number of outputs and also preserves an unique mapping between input and output vectors [15].

Let, the input vector be I_v $\{I_1,\ I_2,\ I_3,\ ...,\ I_n\}$ and the output vector be 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, $I_v \leftrightarrow O_v$. The input vector, I_v and output vector, O_v for 2×2 ***Feynman Gate (FG)*** [16] are $(a,\ b)$ and $(a,\ a \oplus b)$ respectively. Fig. 1 shows the block diagram of Feynman gate and the unique mapping between input-output vectors. The unused outputs of any reversible gate or circuit is called ***Garbage Output*** which will never be used in future rather than to check reversibility [15]. 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. Every reversible circuit has a lower bound of total number of garbage outputs. ***Critical Path Delay*** is the another measurement of the circuit efficiency which is the maximum number of gates from any input to any output [15]. Reversible EX-OR operation requires one gate and the corresponding delay is one.

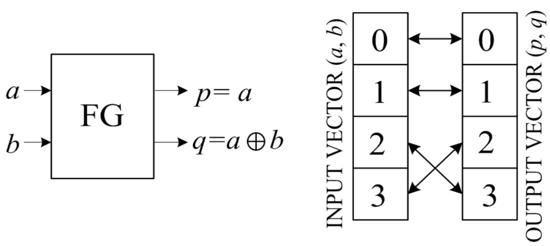

Figure 1: 2×2 Feynman gate and its corresponding input-output mapping

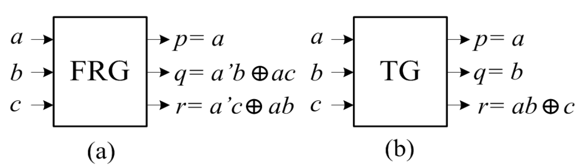

Figure 2: (a) Fredkin gate and (b) Toffoli gate

The input vector, I_v and output vector, O_v of 3×3 ***Fredkin gate (FRG)*** [17] can be defined as: $I_v = (a,\ b,\ c)$ and $O_v = (a,\ a'b \oplus ac,\ a'c \oplus ab)$, respectively. The pictorial representation of FRG is shown in Fig. 2(a). The input and output vector of 3×3 ***Toffoli gate (TG)*** [18] (shown in Fig. 2(b)) are $(a,\ b,\ c)$ and $(a,\ b,\ ab \oplus c)$, respectively.

2.2 Quantum Realization of Reversible Circuit

Quantum Computation is gaining popularity as some exponentially hard problem can be solved in polynomial time [19] and reversibility can be used to construct Quantum circuits [15]. Quantum computation uses matrix multiplication rather than conventional Boolean operations and the information measurement is realized using qubits rather than bits. The matrix operations of qubits are performed by using quantum primitives. The value of qubits is the probability factor of 0 and 1 which are represented as $|0\rangle$ or $|1\rangle$ where

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

The **Quantum Cost (QC)** of any reversible circuit is the total number of 2×2 quantum primitives which are used to form equivalent quantum circuit [15]. The Quantum Cost of reversible Feynman gate (shown in Fig. 3(a)) is 1 because the single 2×2 Quantum EX-OR gate can realize the operation of Feynman gate. Fig. 3(b) shows the quantum circuit representation of Toffoli gate where the quantum cost is 5.

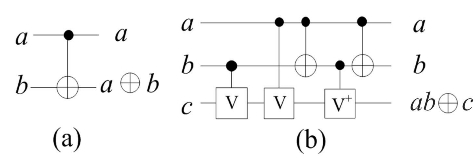

Figure 3: Quantum circuit realization of Reversible gates: (a) Feynman gate (Quantum EX-OR) and (b) Controlled Controlled NOT or Toffoli gate

3. REVERSIBLE PROGRAMMABLE LOGIC ARRAYS

In Section 3.1, we have discussed about the existing design of reversible PLA [13] and its limitations. Rest of the part has described the proposed idea of reversible PLAs based on ordering of output functions and input variables.

3.1 Existing Design of RPLAs

The design of Reversible PLA, is proposed in [13], has used Feynman and Toffoli gates to realize Reversible PLA for multi-output ESOP operation where Toffoli gate is used for AND operation and Feynman gate is used for EX-OR operation. But the existing design has following limitations:

a. Has not treated primary input as garbage when it becomes as an output (But according to [15], unused outputs of any circuits are considered as garbage)

b. Used Conventional Architecture (Complement and non-complement lines for copying input variables)

For example, Equation (I) shows the multi-output ESOP function, where $F = \{f_1,\ f_2,\ f_3,\ f_4,\ f_5\}$. The design used three templates of Feynman gate for implementing COPY, EX-OR and NOT operations (shown in Fig. 4(a)) and single template of Toffoli gate for doing AND operation (shown in Fig. 4(b)). AND plane has used both Feynman and Toffoli gate where as EX-OR plane used only Feynman gate. The realization of multiple-output function in Equation (I) based on existing algorithm is shown in Fig. 4c. The existing design used Toffoli gate AND operation which generates

$$F = \begin{cases} f_1 = ab' \oplus ab'c \\ f_2 = ac \oplus a'b'c \\ f_3 = ab' \oplus ab'c \oplus bc' \\ f_4 = ac \\ f_5 = ab' \oplus ac \oplus bc' \end{cases} \cdots \cdots \cdots (I)$$

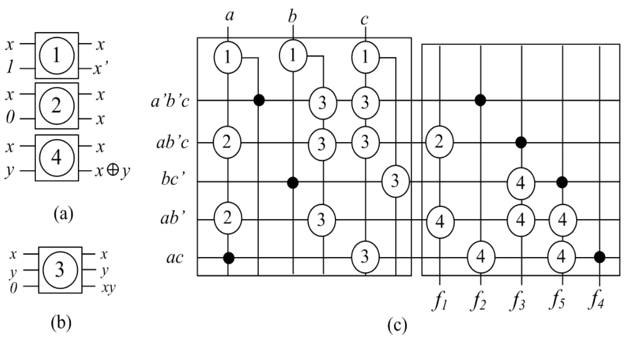

Figure 4: (a) Different templates of Feynman gate for different purposes, (b) Template of Toffoli gate and (c) Design of Reversible PLAs according to [13].

products without changing the form of input literals (complement or non-complement). On the other hand, Toffoli gate produces huge number of unused outputs which are same as primary inputs. But in proposed design of PLAs there is no scope to use those unused outputs.

3.2 Proposed Design of Reversible PLAs

Proposed design is based on the ordering of input variables which depends on the corresponding order of Products. But the order of Products will be generated after the optimization of EX-OR plane. In this subsection, we have proposed two algorithms on the construction of EX-OR plane followed by the realization of AND plane for generalizing the proposed design.The 3×3 reversible MUX [14] or MG gate is used to design proposed Reversible PLAs which has minimum quantum cost i.e 4 (shown in Fig. 5). MG gate can realize the operation of (2 to 1) multiplexer circuit and able to produce half of minterms generated by two variables. Fig. 6 shows the templates (MG-5 and MG-6) of a MUX gate which have been used in proposed design. We have used three modes (FG-1, FG-2 and FG-4) of Feynman gate proposed in [13] (shown in Fig. 4(a)) and other two modes (MG-5) and (MG-6) of MUX gate shown in Fig. 6. In our further discussion, we have used symbol 1, 2, 4 (Fig. 4) and symbol 5, 6 (Fig. 6) rather than full name (FG-1 or MG-5 etc.) to represent the particular modes. The cross point in RPLA, in which no gate is used, is termed as DOT [13].

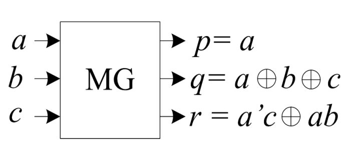

Figure 5: 3×3 Reversible MUX gate

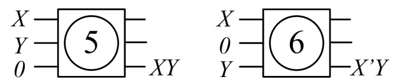

Figure 6: Two different templates of MUX gate

Table 1: Size of each function of Equation (I)

function	f_1	f_2	f_3	f_4	f_5
SizeOf (f_i)	2	2	3	1	3

Definition 1. **Size of Function (SizeOf (f_i))**: Total number of products has been used by f_x function. For example, in Equation (I), SizeOf(f_1)= 2 because the total number of products of f_1 is 2 (shown in Table 1).

Definition 2. **Product Lines** are the horizontal lines corresponding to the products of an AND plane. These product lines are used in the EX-OR plane to generate the output of a particular function consisting of EX-OR operations. The number of product lines is equal to the number of total products. There are 5 product lines for 5 products of function F (see Equation (I)).

In the proposed design, the ordering of output functions is related to SizeOf (f_x). Functions are generated in ascending order based on this criterion. We have optimized EX-OR Plane followed by AND Plane minimization by using MUX and Feynman gates.

Algorithm 1: Construction of EX-OR Plane

1. **START** $TDOT$:= 0 [$TDOT$= Total number of DOT]
2. Sort output Functions according to Sizeof (f_i)
3. **REPEAT** Step 4 for each output function
4. **IF** Sizeof (f_i) of f_i is one **THEN**
5. **IF** product p_j exists **THEN** use FG-2
6. **ELSE** assign a line for product (p_j) and use DOT
7. $TDOT$:= $TDOT$+ 1
8. **END IF**
9. **ELSE**
10. **IF** all product(s) p_j exist **THEN** use FG-2 for top most line and FG-4 for others
11. **ELSE** assign the upper lines for products p_j and use DOT for top most and FG-4 for existing
12. $TDOT$:= $TDOT$+1
13. **END IF**
14. **END IF**
15. **END**

By using the proposed algorithm, the realization of EX-OR plane for Equation (I) is shown in Fig. 7.

THEOREM 1. *Let n be the number of EX-OR operations of m output functions and $TDOT$ be the number of cross-points, then the minimum number of Feynman gates to realize EX-OR plane is $n + m - TDOT$.*

PROOF. When there are $TDOT$ cross-points for m functions, the number of additional Feynman gates in EX-OR plane of RPLA is $m - TDOT$. As there are n Ex-OR operations by n Feynman gates, the total number of Feynman gates in the EX-OR plane of RPLA is $n + m - TDOT$. □

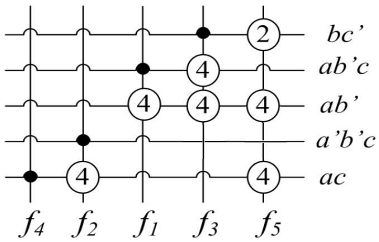

Figure 7: EX-OR plane realization of Equation (I) based on proposed Algorithm 1

For multi-output function F in Equation (I), the number of outputs (m) is 5 and the number of EX-OR operations is 6. The number of $TDOT$ is 4 in Fig. 7. So the number of Feynman gates= n + m - $TDOT$ = 6 + 5 - 4 = 7.

Theorem 2. *Let p be the number of products and $TDOT$ be the number of cross-points in the EX-OR plane of RPLA. The minimum number of garbages to realize EX-OR plane of RPLA is $p - TDOT$.*

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 $p - TDOT$. □

Consider Fig. 7 for multi-output function F in Equation (I). In Fig. 7, number of products (p) is 5 and number of cross-points $(TDOT)$ is 4. So the number of garbage outputs is p - $TDOT$ = 5 - 1 = 4. The realization of EX-OR plane generates the order of Products shown in Fig. 7 and AND plane will be constructed according to this order by using MUX and Feynman gates.

Algorithm 2: Construction of AND Plane

1. **START** $TDOT$:= 0 [$TDOT$= Total number of DOT]
2. **REPEAT** Step 3 for each product (p_j)
3. **IF** l_j is the first literal of p_j **THEN**
4. **IF** l_j is in complemented form THEN apply FG-1
5. **ELSE**
6. **IF** l_j is further used **THEN** apply FG-2
7. **ELSE** use DOT and $TDOT$:= $TDOT$ + 1
8. **END IF**
9. **END IF**
10. **ELSE IF** l_j in complemented form **THEN** apply MG-6
11. **ELSE** apply MG-5
12. **END IF**
13. **END**

In AND plane, the Feynman gates are used to copy or recover fan-out problem and the MUX gates are used for AND operations. The generation of complementary forms of input literals are unnecessary for the proposed AND plane because MUX and FG are used together to generate all the minterms of two variables without having any dedicated lines of complemented forms of input variables. By using Algorithms 1 and 2, the realization of proposed Reversible PLA is shown in Fig. 8.

Theorem 3. *Let p be the number of products and $TDOT$ be the number of cross-points in the AND plane of RPLA. The minimum number of Feynman gate to realize AND plane of RPLA is $p - TDOT$.*

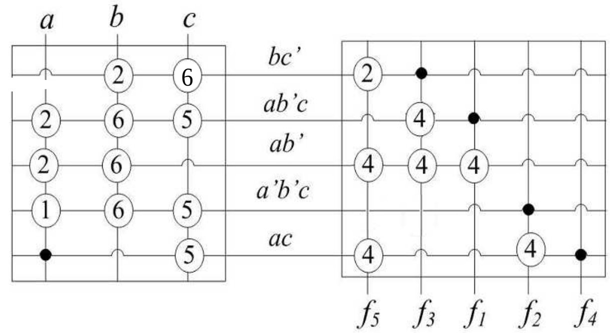

Figure 8: Proposed reversible PLAs design of multi-output function F in Equation (I)

Proof. Cross-points reduce the usability of Feynman gates for any particular product line. So, in response to $TDOT$ cross point and p products for m output functions, the total number of Feynman gates in the AND plane of RPLA is $p - TDOT$. □

Consider Fig. 8 for multi-output functional Equation (I). In Fig. 8, number of products (p) is 5 and number of cross points $(TDOT)$ is 1. So the number of Feynman gates is p - $TDOT$= 5 - 1 = 4.

Theorem 4. *Let* q *be the number of AND operations among products in the AND plane of RPLA. The minimum number of MUX gate to realize AND plane of RPLA is* q.

Proof. As there are q AND operations for p products of m output functions, the total number of MUX gates to realize the AND plane of RPLA is q. □

Theorem 5. *Let l be the number of inputs and q be the number of AND operations among products and $TDOT$ be the number of cross-points in the AND plane of RPLA. The minimum number of garbages to realize AND plane of RPLA is $l + q - TDOT$.*

Proof. As there are q AND operations for p products of m output functions F in Equation (I), l inputs and $TDOT$ cross-point, the total number of garbages in the AND plane of RPLA is $l + q - TDOT$. □

3.3 Delay Calculation

In this paper, we have calculated the delay of reversible PLAs in greedy approach and the proposed algorithm generates better throughput. We divide the calculation into two phases: a. AND Plane Delay (APD (p_i)) and b. EX-OR Plane Delay (XPD (p_i)) in terms of product lines (horizontal lines). Then we have merged both of the delay respect to both planes. We have used Equation (I) to calculate the delay. First we calculate the delay of AND plane followed by EX-OR plane. Fig. 9 and Fig. 10 show the delay calculation of AND plane and EX-OR plane respectively. In further realization of delay calculation, we consider the following things:

a. Gate (Via) is represented as circle (DOT).

b. Delay of any gate is 1 and via (DOT) denotes 0.

c. Decimal values show the delay of corresponding circle

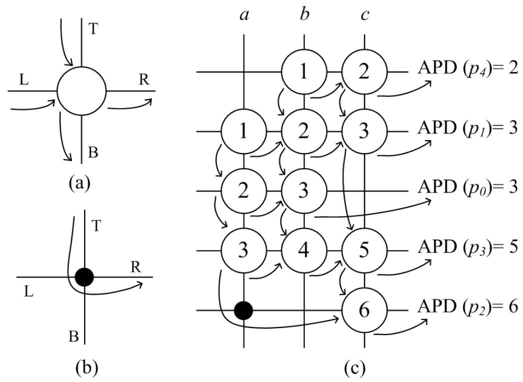

Figure 9: Delay calculation of AND plane: (a-b) Delay propagation path of a gate and a cross-point respectively in AND plane and (c) Overall delay propagation path for AND plane

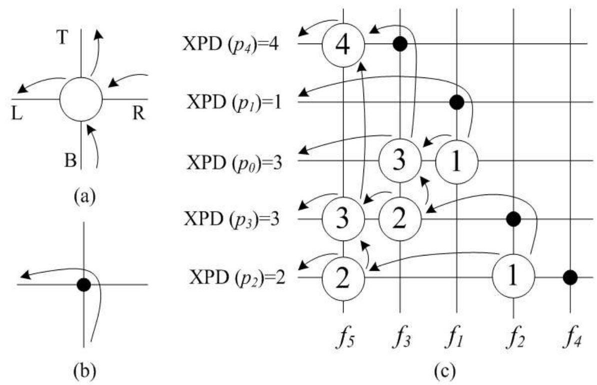

Figure 10: Delay calculation of EX-OR plane: (a-b) Delay propagation path of a gate and a cross-point respectively in EX-OR plane and (c) Overall delay propagation path for EX-OR plane

3.3.1 Delay Calculation of AND Plane

For AND plane, every gate updates its Delay by comparing the delay of neighboring gates at Left (L) and Top (T) and then, it propagates the updated delay to neighboring gates placed in Right (R) and Bottom (B) sides as shown in Fig. 9. Each Circle in Fig. 9 represents the delay of particular point and Arrows show the path of delay propagation. The size of AND plane of proposed design is less then existing design because proposed design does not need to generate complement lines of corresponding input lines.

3.3.2 Delay Calculation of EX-OR Plane

For EX-OR plane, every gate updates its Delay by comparing the delay of neighboring gates at Right (R) and Bottom (B) and then, it propagates the updated delay to neighboring gates placed in Left (L) and Top (T) sides as shown in Fig. 10. The size of EX-OR plane of proposed design is same as existing design and this plane is optimized in terms of cost analysis as Theorem 1 & 2.

3.3.3 Delay of Overall Design

After calculating the delay of both planes, the delay of product lines having maximum value is the final delay of the overall design of reversible PLAs. We proposed the following algorithm for the calculation of delay of reversible PLAs. According to proposed design, the delay propagation of AND (EX-OR) plane is Top-Bottom-Right (Bottom-Top-Left).

Algorithm 3: Delay Calculation of Reversible PLAs

---

1. **START** Calculate APD (p_i) (XPD (p_i)) of each product lines of AND (XOR) plane
2. Delay:= MAX {APD (p_i) + XPD (p_i)} where i= 1 to n (n= total number of product)
3. **END**

3.4 Experimental Results

We have realized the calculation of all proposed algorithms by using programming language Java (J2SE 1.6.0_17) on Netbeans IDE (6.8) in Linux Workstation. All the experimental results are tested on Intel(R) Core(TM)2 Duo CPU E7300 2.66GHz edition with 2 GB RAM. Table 2 shows the experimental results for different benchmark functions and the comparison with the existing method [13].

4. CONCLUSIONS

In this paper, we proposed a regular structure of Reversible Programmable Logic Arrays (RPLAs) based on MUX-Feynman logic and also we presented the minimization techniques for both AND and EX-OR planes of reversible PLAs. We used the garbage outputs as operational outputs that reduced the number of AND operations in RPLAs. The minimization of AND plane based on the ordering of input variables gives an excellent throughput of the overall design. Finally, we figured the performance of the proposed design over the existing one. The experimental results show that the proposed design outperforms the existing one in terms of numbers of gates, garbages and quantum costs. The proposed algorithm also required less time than the existing one. We also presented five lower bounds on the numbers of gates, garbages and quantum cost of RPLAs. RPLAs are useful in embedded circuits and others technologies for power consumption and fault tolerant [8], [4], [18].

5. ACKNOWLEDGMENTS

We would like to thank Mr. A. R. Chowdhuary, Assistant Prof. of Department of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh. He is currently pursuing PhD at the Monash University, Australia.

6. REFERENCES

[1] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Research and Development*, 5(3):183–191, 1961.

[2] C. H. Bennett. Logical reversibility of computation. *IBM J. Research and Development*, 17:525–532, 1973.

[3] I. Hashmi and H. M. H. Babu. An efficient design of a reversible barrel shifter. In *IEEE 23rd International Conf. on VLSI Design*, pages 93–98, Banglore, India, 2010.

[4] G. Schrom and S. Selberherr. Ultra-low-power cmos technology. In *Semiconductor Conf.*, Romania, 2002.

Table 2: Experimental results using different benchmark functions

	Proposed Design					Existing Design [13]				
	*GA	*GB	*DL	*QC	*CT[ms]	GA	GB	DL	QC	CT[ms]
5xp1	166	112	36	418	1.689	170	118	30	508	3.648
9sym	427	385	59	1405	0.844	439	391	62	1737	1.305
adr3	67	48	19	157	0.153	69	50	19	189	0.268
apex3	3998	1799	279	8654	4.376	4047	1848	234	10255	8.678
b12	159	132	35	453	0.499	170	140	29	562	0.652
bw	305	64	44	446	0.499	350	70	45	499	0.572
cordic	12162	10640	792	41694	19.737	18184	10662	792	51560	75.955
duke2	941	667	99	2735	1.305	931	695	93	3361	1.804
e64	2170	2148	114	8410	2.572	2228	2206	116	10548	3.072
inc4	16	10	9	34	0.192	16	10	6	40	0.153
inc5	23	16	8	53	0.153	24	17	8	64	0.115
misex1	88	51	22	193	0.192	95	58	20	235	0.192
misex2	199	176	36	622	0.384	227	200	34	787	0.460
pdc	3801	3006	275	12096	7.372	3825	3030	273	14885	13.939
rd53	56	42	17	134	0.152	55	44	18	162	0.192
rd73	187	141	44	487	0.345	188	144	37	590	0.499
rd84	328	265	68	928	0.844	321	269	58	1132	1.382
sao2	284	236	41	890	0.691	291	243	41	1099	0.768
sasao	14	13	9	29	0.115	15	14	8	35	0.115
t481	49	53	17	133	0.192	64	68	17	176	0.230
table3	2602	1814	190	7537	2.956	2619	1831	173	9199	4.569
table5	2539	1819	182	7516	2.841	2559	1839	148	9195	4.416
vag2	2018	1836	188	6926	3.955	2038	1856	173	8582	7.680
xor5	8	8	5	8	0.115	9	9	5	9	0.075
z5xp1	167	114	35	425	0.345	171	122	29	519	0.422
z9sym	427	385	59	1405	0.960	433	391	62	1737	1.459

*GA= Total Gates; *GB= Total Garbages; *QC= Quantum Cost; *DL= Delay of the Circuit; *CT= CPU Time in millisecond (ms)that required to calculate the numbers of gates and garbages as well as delay counting and quantum cost analysis.

[5] E. Knill, R. Laamme, and G. J. Milburn. A scheme for e-cient quantum computation with linear optics. *Nature*, 409:46–52, January 2001.

[6] M. A. Nielsen and I. L. Chuang. Quantum computation and quantum information, 2001.

[7] R. C. Merkle. Two types of mechanical reversible logic. *Nanotechnology*, 4(2):114–131, January 1993.

[8] H. Fleisher and L. I. Maissel. An introduction to array logic. *IBM J. of Research and Development*, 19, 1975.

[9] T. Sasao. Exmin2: A simplification 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*, 12(5):621–632, 1993.

[10] A. Mishchenko and M. Perkowski. Logic synthesis of reversible wave cascades. In *International Workshop on Logic Synthesis*, pages 197–202, June 2002.

[11] M. Perkowski, A. B. P. Kerntof and et al. Regularity and symmetry as a base of efficient realization of reversible logic circuits. In *International Workshop on Logic Synthesis*, pages 245–252, June 2001.

[12] D. Maslov and G. Dueck. Reversible cascades with minimal garbage. *IEEE Transactions on CAD*, 23(11):1497–1509, November 2004.

[13] A. R. Chowdhury, R. Nazmul, and H. M. H. Babu. A new approach to synthesize multiple-output functions using reversible programmable logic arrays. In *IEEE 19rd International Conf. on VLSI Design*, pages 311–316, Hyderabad, India, 2006.

[14] A. S. M. Sayem and S. K. Mitra. Efficient approach to design low power reversible logic blocks for field programmable gate arrays. In *IEEE International Conference on Computer Science and Automation Engineering*, pages 251–255, July 2011.

[15] A. K. Biswas, M. M. Hasan, A. R. Chowdhury, and H. M. H. Babu. Efficient approaches for designing reversible binary coded decimal adders. *Elsevier Microelectron. Jounrnal*, 39(12):1693–1703, December 2008.

[16] R. P. Feynman. Quantum mechanical computers. *Opt. News*, 11(2):11–20, 1985.

[17] E. Fredkin and T. Toffoli. Conservative logic. *International J. Theoretical Physics*, 21:219–253, 1982.

[18] T. Toffoli. Reversible computing. *MIT Lab for Comp. Sci.*, 85:632–644, 1980.

[19] V. V. Shende, A. Prasad, I. Markov, and J. Hayes. Reasoning about naming systems. *IEEE Trans. CAD*, 22(6):723–729, 2003.