Reversible Programmable Logic Arrays

Sajib Kumar Mitra
Dept. of Computer Science and Engineering
University of Dhaka
Dhaka-1000, Bangladesh
Email: sajibmitra.csedu@gmail.com

Md. Riazur Rahman
Dept. of Computer Science and Engineering
Daffodil International University
Dhaka-1205, Bangladesh
Email: riazur_rahman@daffodilvarsity.edu.bd

***Abstract*—An novel approach to design Reversible PLAs maximizing the usability of garbage outputs and reducing the number of ancilla inputs is presented. A brief overview of proposed fundamental components and the architecture of reversible grid network for designing AND plane are then presented. Several algorithms have been used to describe the programming interfaces in context of Reversible PLAs construction. Finally, recent result on the trade-off between cost factors of standard benchmark circuits is reviewed.**

I. Introduction

During execution of every single instruction, stuff wastes $kT\ln(2)$ joule of energy as converted into heat due to per bit erase and reload where T is the operating temperature and k is the Boltzmann constant [1]. Solution of such input energy loss mechanism after publishing the tremendous approach called Reversible Computation was introduced by Bennett [2] in 1973. It opens the tunnel of designing robust architecture of low power consumption where total input energy loss is zero, supports the behavior of optical computing, quantum computing, etc. Generic prototype of designing low power programmable devices [3] has obtained popularity in recent days. So, the development of reversible PLAs would be another application that enhances low power computing. Proposed idea presents the novel architecture of PLAs in reversible computing by attaching 100% uses of every logical units/gates propagates all primary inputs to outputs. Proposed architecture reflects the following ideology:

- Maximize the usability of primary input signals
- Avoiding any type of EX-OR operations in AND plane
- Reduce number of garbage outputs and ancillia inputs

Rest of the paper has been organized as: section II has described reversible logic and the standards of measuring the performance of reversible circuits. Section III has presented the details of proposed gates and demonstrate organizational placement of logical units (gates) in Reversible PLAs grid. Section IV has illustrated the corresponding algorithms for constructing AND and EX-OR planes using reversible {UMG and UNG} and CNOT gates, respectively. Comparative performance analysis based on benchmark standard circuits has been showed in section V. Finally, section VI has concluded this paper with the summary and future directions.

II. Background Study

A. What is Reversible Gates?

Bidirectional or reversible circuit prevents input loss due to unique mapping between input and output states. Like classical computing, any reversible operational unit entity is called $n \times n$ Reversible gate contains:

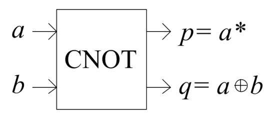
- n-input lines and n-output lines
- Unique mapping between input and output states

For example, controlled NOT (CNOT), widely known as Feynman gate [4] is reversible has two inputs (a, b) and two outputs (p, q) is shown in Fig. 1. Total number of input and output states are same (i.e. 4) and the mapping between input and output states is unique or vice versa.

There are many reversible gates have been populated based on conservative logic [5], universality of reversible circuit [6], fault tolerant mechanism [7], online testability [8], programmable devices [9], etc. Several reversible gates are self-reflexive backs primary input signals by attaching self-copy and other gates stuck signals need extra circuitry to return in initial state. In this paper, two new 3×3 reversible gates called Universal MUX (UMG) and Universal NOR (UNG) are used to design AND plane of reversible PLAs where UNG is self-reflexive reversible gate performs basic OR (or universal NOR) operation and returns primary inputs to output. On the other hand, UMG also performs AND operation and returns primary inputs as like UNG but not self-reflexive.

B. Performance Measurement Standards

Operational Competency of any circuit is always related to its technical design encroachment. In any particular technology, greater number of logical units slows down the strength of signal hampers net processing speed of circuit. But interestingly, logical minimization provides better opportunity to reduce the number of operational units and total cost.



Input		Output	
a	b	p	q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Fig. 1. Reversible CNOT and unique IO states mapping

1) Total Number of Gates: In reversible circuit, the input loss is zero in ideal state but bending input signals to output lines absorbs energy and declines the strength of internal signal due to have unavoidable resistance. The total number of gates used in circuit is considered one of the worthy cost factor controls performance of digital circuit [10].

2) Quantum Cost: Every reversible circuit points unique singular unitary matrix which can be accomplished with one or more 2×2 and 4×4 unitary matrices whose are also compatible to 1×1 and 2×2 basic primitives in Quantum Computing. Alternatively, the n-dimensional quantum primitive is identically formed of $2^n\times2^n$ dimensional unitary matrix. The total number of 2×2 quantum primitives are used to realize any reversible circuit is called Quantum Cost [11].

3) Garbage Output and Ancilla Input: Unlike classical computing, reversible circuit requires extra output lines to map all input the states uniquely, called garbage output [12]. On the other hand, one or more input line(s) get saturated in constant level (i.e. 0 or 1) to perform specific operations is called Ancilla Input [13].

According to above definitions, the realization of 2-input EX-OR operation requires only one 2×2 reversible Feynman gate and the quantum cost of Feynman gate is 1 (single 2×2 quantum XOR gate is able to realize CNOT operation), the number of garbage output is 1 and finally, the number of ancilla inputs is zero (shown in Fig. 1).

C. Review on Reversible PLAs

In 2006, author of [14] has proposed the Reversible architecture of PLAs that was similar to classical PLA design [3] where AND plane consists of vertical complement and non-complement input lines and horizontal products lines spread over EX-OR plane. Toffoli gates were used to perform AND operation in AND plane whereas Feynman performed EX-OR operation in EX-OR plane. Additionally, Feynman gates were also used in AND plane for avoiding fan-out(s). The improved design of [14] was proposed in [15] brought prominent modification in the basic architecture of classical reversible PLA circuits by using only single line for each input literal in AND plane. Ref. [15] used MUX and Feynman gates to realize improved design of reversible PLAs where AND plane also performed copy operation by using Feynman gates as the similar way in [14]. Both papers had used multiple output functions F (i.e. *Eqn. 1*) as a sample to represent their proposed designs and minimization methodologies.

$$F = \left\{\begin{array}{c} f_1 = ab' \oplus ab'c \\ f_2 = ac \oplus a'b'c \\ f_3 = ab' \oplus ab'c \oplus bc' \\ f_4 = ac \\ f_5 = ab' \oplus ac \oplus bc' \end{array}\right\} \cdots \quad \cdots \quad \cdots (\textit{Eqn. 1})$$

D. Motivation of this Research Work

Fundamentally, classical architecture of PLAs [3] was implemented by placing configurable switches at cross-point. These switches copy input signal multiple times increases fan-outs (which is restricted in Reversible Computing). The simplicity of AND, OR and NOT logic has been promoted by novel researchers to design such architecture of Programmable Logic Devices (PLD) in classical digital circuit [3]. But in reversible computing, the operability of basic bidirectional components is unavoidable when designing logic circuit such as Reversible Programmable Logic Arrays. In both [14] and [15], reversible PLAs focused the ideal zero energy dissipation due to use of large number of CNOT gates to recover fan-out(s) increases number of ancilla bits and garbage outputs. The idea of proposed research work comes through the reusability of garbage outputs as the input to next operational unit(s) that reduces the number of ancilla input at the same time.

III. Proposed Reversible Gates and PLA Grid

In this section, two new reversible primitives (Universal MUX and Universal NOR gates) have been introduced followed by the demonstration of logical units placement in AND plane as well as the ordering principle of products generation.

A. New Reversible Gates and Operational Templates

1) Reversible Universal MUX Gate (UMG): The input and output vectors of Universal MUX gate can be written as (a, b, c) and ($p= a$, $q= b\oplus c$, $r= ab\oplus a'c$), respectively. The equivalent quantum representation of UMG is shown in Fig. 2.

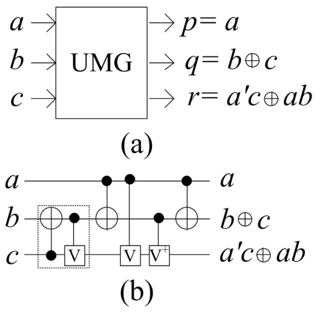
2) Reversible Universal NOR Gate (UNG): In Boolean logic, NOR gate is an universal primitives can interpret the functionalities of all basic gates (AND, OR, NOT). Similarly, the input and output vectors of proposed 3×3 Universal NOR gate which perform NOR operation can be written as (a, b, c) and ($p= a$, $q= b$, $r= (a+b)\oplus c$), respectively (shown in Fig. 4).

Proposed UMG and UNG gates are used to perform AND operation of two literals (or a literal and a product). The forms of logical unit(s) which are used in proposed reversible PLAs have been selected based on following facts:

- Best orientation of Input and/or Output line(s)
- Projected output(s)(product/sum) of plane(AND/OR)

UMG performs MUX operation by setting input a as selection line and others (b and c) as data. Proposed UMG is able to generate three minterms of two inputs (ab, ab' and $a'b$, are represented through templates α, β and γ (swapping orientations are α', β' and γ') as shown in Fig. 3. UMG doesn't erase the input value of any operational unit while performing AND operation and those unused outputs can be used as the primary inputs to another reversible gates. UNG recovers the limitation of UMG and the operational template is symbolized using π (π') (shown in Fig. 5).

Algorithm 1 shows the methodology of selecting template (oriented form of logical unit) to perform AND operation of inputs p and q depending on the value of *swapFlag*. The statement, *swapFlag* = 0 indicates to perform AND operation by using α', β' or γ' (otherwise α, β, γ or π).



Input			Output		
a	*b*	*c*	*p*	*q*	*r*
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	0	0	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	1	0	1

Fig. 2. Reversible Universal Multiplexer gate (UMG): (a) Block diagram of UMG and (b) Quantum realization of UMG; Truth table of UMG maps uniquely all the input states to output states

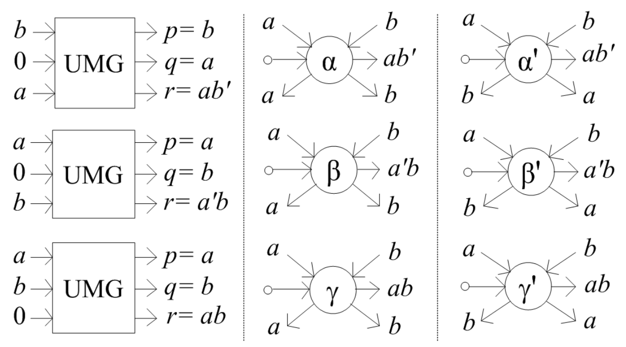


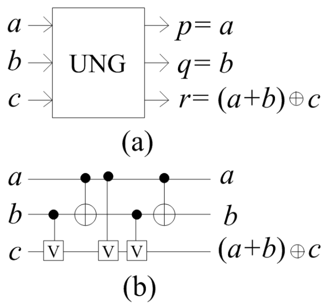
Fig. 3. Proposed templates of Universal MUX gate (UMG) are used in proposed Reversible PLAs design are symbolized through α, β and γ whereas the swapping templates are α', β' and γ'

Algorithm 1: OpAND(p, q, *swapFlag*)

Templates $\{\alpha, \beta, \gamma, \pi\}$ (for swapping $\{\alpha', \beta', \gamma', \pi'\}$) are used to AND $\{p, q\}$ based on the value of *swapFlag*.

```
Start
1.  If $p$ is a Literal in complemented form Then
2.      If $q$ is in complemented form Then
3.          If $swapFlag = 0$ Then use $\pi'$ Else use $\pi$
4.          End If
5.      Else
6.          If $swapFlag = 0$ Then use $\beta'$ Else use $\beta$
7.          End If
8.      End If
9.  Else
10.     If $q$ is in complemented form Then
11.         If $swapFlag = 0$ Then use $\alpha'$ Else use $\alpha$
12.         End If
13.     Else
14.         If $swapFlag = 0$ Then use $\gamma'$ Else use $\gamma$
15.         End If
16.     End If
17. End If
18. Return $p.q$
End
```

On the other hand, proposed EX-OR plane consists of only Feynman gates are used to perform XORing products. Three



Input			Output		
a	*b*	*c*	*p*	*q*	*r*
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	1	0

Fig. 4. Reversible Universal NOR gate (UNG): (a) Block diagram of UNG and (b) Quantum cost is 5; UNG maps input and output states uniquely

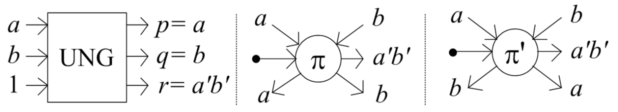


Fig. 5. Templates of UNG are used to generate product (a′b′)

templates of Feynman gates have been used in proposed design are symbolized through symbols $\triangle$, λ and ∇ perform NOT $\{a, a'\}$, EX-OR $\{a, (a \oplus b)\}$ and COPY operation $\{a, a\}$, respectively (shown in Fig. 6).

B. Reversible PLAs Grid and Primitives Placement

Reversible gates are more powerful performs multiple logic operations in single cycle [10]. The orientation of input and product lines of proposed AND plane is pointed through solid lines (shown in Fig. 7a) where dotted lines indicate another pathway to swap input signals (shown in Fig. 7b) according to following algorithm (Algorithm 2).

Algorithm 2: SwapLiterals(L_i, L_j)

Exchanging Input signals $\{I_i, I_j\}$ in lines, $\{L_i, L_j\}$.

```
Start
1. Set $a$:= signal at input line, $L_i$
2. Set $b$:= signal at input line, $L_j$
3. $L_i$:= $b$ and $L_j$:= $a$
End
```

Basically, swap operation of two literals be performed when the uses of any literal got ended for doing AND operation in AND plane. **SwapLiteral(*L_i*, *L_j*)** moves unused literals from left to right vertical tracks of AND Plane. Performing AND operation at any cross-point of two vertical lines binds single horizontal line to generate cumulative product and again, connecting another literals to cumulative product (if needed) to generate final product of AND plane.

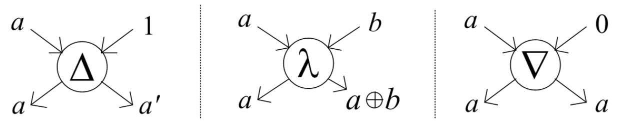


Fig. 6. Templates of CNOT gate are used to design EX-OR plane

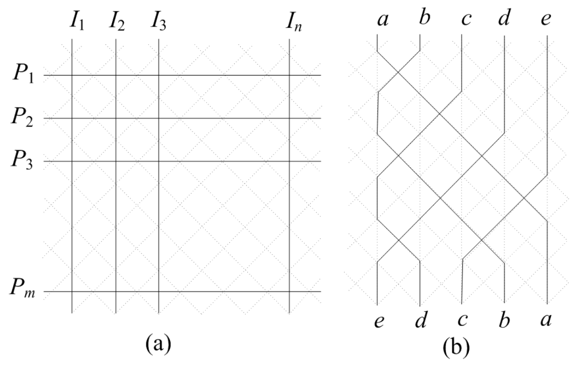


Fig. 7. Reversible PLAs Grid Architecture of AND plane: (a) Primary inputs and generated products be passed through vertical and horizontal lines, respectively and (b) Swapping input signals

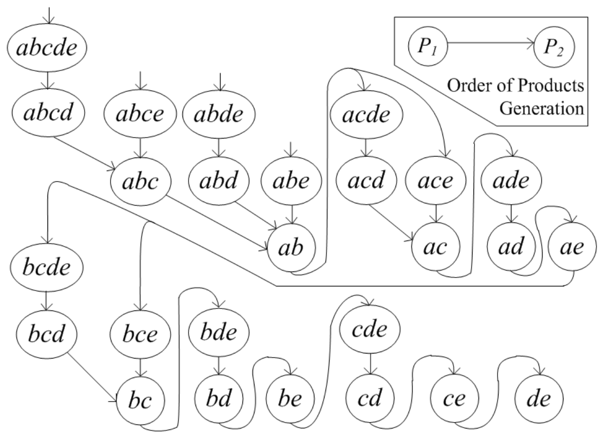


Fig. 8. The order of products consist of literals (*a*, *b*, *c*, *d*, *e*) is: {[*abcde*, *abce*, *abde*, *abe*], [*abcd*],[*abc*, *abd*], [*ab*], [*acde*, *ace*], [*acd*], [*ac*], [*ade*], [*ad*], [*ae*], [*bcde*, *bce*], [*bcd*], [*bc*], [*bde*], [*bd*], [*be*], [*cde*], [*cd*], [*ce*], [*de*]}. Products in the same group (for example [*abcde*, *abce*, *abde*, *abe*]) are independent can be generated in any order.

Ordering products takes crucial role reduces the cost of products generation by using garbage output(s). Also, the usability of different templates provides mining opportunities optimizing the cost in physical layer. Resultant products contain same number of literals placed according to the order of input literals. For example, products (P_v) consist of literals $\{a$, b and $c\}$ be produced in order, product(s) start with *a* followed by start with *b* followed by start with *c*. Algorithm 3 describes the methodology of placing products based on their useability. For example, product *abcd* be generated before *abc*, *abd* or *ab* which are consisted of less number of literals.

Algorithm 3: OrderingProducts(I_v, P_v)

```
Products, $P_v$ be ordered according to inputs, $I_v$.
Start
1. Set $P_Q$:= $\phi$ [$P_Q$ is used to store products]
2. Sort $P_v$ based on SizeOf($P_i$) in descending order
3. For $i$= 1 to totalLiterals
4.     For $j$= 1 to totalProducts
5.         If $I_i \in P_j$ and $P_j \notin P_Q$ Then Add $P_j$ to $P_Q$
6.         End If
7.     End Loop
8. End Loop
9. Set $P_v$ := $P_Q$
End
```

According to above algorithm (ALG. 3), the order of products consists of inputs *a*, *b*, *c*, *d* and *e* (only the non-complemented forms) can be graphed as shown in Fig. 8.

IV. Proposed Design of AND and OR Planes

In this section, proposed design of AND plane based has been described followed by the realization of EX-OR plane.

A. Designing Reversible AND and EX-OR Planes

AND plane dominates the performance and cost factors of reversible PLAs where every AND operation rises all cost factors compared to simple of EX-OR operation. Algorithm 4 presents the construction of proposed AND plane as well as the minimization of garbage outputs. The construction of AND plane includes, the ordering of products (Algorithm 3) followed by counting *swapFlag* and then invokes **OpAND(*p*, *q*, *swapFlag*)** (Algorithm 1). Input lines exchange signals (as Algorithm 2) after finishing the generation of all mutual product(s). Finally, **Queue (P_{QG})** stores unused garbage products which are used in afterward as resultant products when they get similar to unexplored products.

Algorithm 4: ConstuctANDPlane(I_v, P_v)

```
This function constructs AND plane by taking set of input literals ($I_v$) and generates products ($P_v$) connecting multiple input lines ($L_v$) by using UMG and UNG gates.
Start
1.  OrderingProducts ($I_v$, $P_v$)
2.  Set $P_{QG}$:= $\phi$ and ndot:= 0 [$P_{QG}$ stores garbage]
3.  For $g$ = 1 to totalLiterals - 1
4.   For $h$ = $g$ + 1 to totalLiterals
5.    Set swapFlag:= 0
6.    For $i$ = 1 to totalProducts
7.     If $I_gI_h \in P_i$ Then swapFlag:= swapFlag + 1
9.     End If
10.   End Loop
11.   If swapFlag > 0 Then
12.    For $i$ = 1 to totalProducts
13.     If SizeOf($P_i$) > 1 Then
14.      If $P_i \in P_{QG}$ Then Remove $P_i$ from $P_{QG}$
15.      Else
16.       If $I_gI_h \in P_i$ Then swapFlag:= swapFlag - 1
17.        Set pivotP:= OpAND($I_g$, $I_h$, swapFlag)
18.       If SizeOf($P_i$) > 2 Then
19.        For $j$ = $h$ + 1 to totalLiterals
20.         If $I_j \in P_i$ Then
21.         Set $P_g$:= pivotP
22.          pivotP:= OpAND(pivotP, $I_j$, false)
23.         End If
24.        End Loop
```

```
25.             Add $P_g$ to $P_{QG}$ [Add new garbage to $P_{QG}$]
26.           End If
27.         End If
28.       End If
29.     Else $ndot$:= $ndot$ + 1 [Use via (●)]
30.     End If
31.   End Loop
32.   Else SwapLiterals($L_g$, $L_h$); [No mutual products]
33.   End If
34.  End Loop
35. End Loop
End
```

Theorem 1. *Let, n be the number of AND operations of m output functions and t be the number of AND operation of garbage outputs,* (P_{QG}) *which are identical to any products then the minimum number of reversible gates to realize AND plane is (n-t), the quantum cost is 5(n-t), the number of ancilla input is (n-t).*

Proof: As performing every reversible AND operation needs single UMG or UNG gate, results total number of gates to realize AND plane is n. But reusability of garbage reduces the number of acting AND operations is to (n-t). Similarly, the quantum cost of UMG or UNG is 5 sums-up the total quantum cost of circuit is 5(n-t) and every reversible AND operation requires an ancilla bit summarizes total number of ancilla inputs to (n-t). ■

For multi-output function F in Eqn. (I), total number of AND operations (n) is 7, the number of AND operation(s) in garbage which are similar to any product (t) is 1. So, the number of gates= (n-t)= 7-1= 6, quantum cost= 30 and total ancilla input= 6 (shown in Fig. 9).

Theorem 2. *Let, p be the number of products (consist of more than two literals) of m output functions, q be the number of garbage outputs which are identical to any products and ndot be the number of cross-point then the number of garbage is p+totalLiterals-ndot-q.*

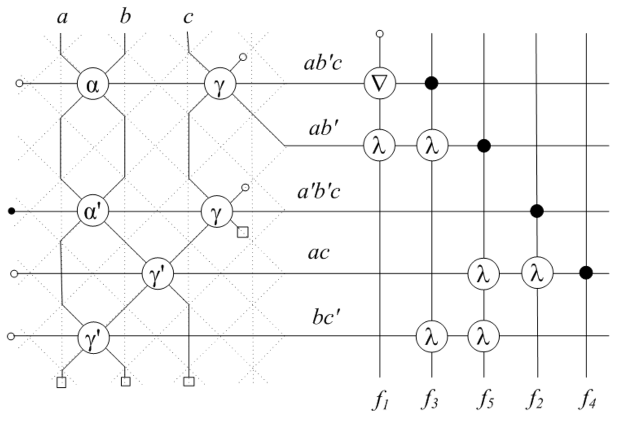


Fig. 9. Optimized version of reversible PLAs of multi-output function F in Equation(I)

Algorithm 5 describes the construction of EX-OR plane by using Feynman gates where (λ) connects product lines (P_i) to corresponding function lines (F_j) produces output signals and another identical copy of P_i.

Algorithm 5: ConstructXORplane(P_v, F_v)

EX-OR plane generates the final outputs of multi-output function, (F_v) consists of products, P_v.

```
Start
1.  Set $F_Q$:= $\phi$ and $xdot$:= 0
2.  For $i$= 1 to $totalProducts$
3.    For $j$= 1 to $totalFunctions$
4.      If $P_i \in F_j$ Then
5.        If $F_i \notin F_Q$ Then
6.          If FreqOf($P_i$) == 1 Then
7.            $xdot$:= $xdot$ + 1 [use via (●)]
8.          Else
9.            Use ∇ [Keep a copy of $P_i$]
10.           Set FreqOf($P_i$):= FreqOf($P_i$) - 1
11.         End If Add $F_i$ to $F_Q$
12.       Else Use $\lambda$ [XORing $P_i$ to $F_j$ Line]
13.       End If
14.     Else If $P_i' \in F_j$ Then
15.         Use △ [Keep a copy of $P_i$]
16.     End If
17.   End Loop
18. End Loop
End
```

Theorem 3. *Let, n be the number of EX-OR operations of m output functions and xdot be the number of cross-points, then the minimum number of Feynman gates to realize EX-OR plane is n+m-xdot, the quantum cost is n+m-xdot, total number of ancilla input is m-xdot.*

According to proposed algorithms (ALG. 4 & 5), the construction of multi-output function F in Equation (I) is shown in Fig. 9 where garbage outputs are represented by using line ends with box. Table I summarizes that the proposed design of reversible PLAs requires less number of gates, garbage outputs and ancilla inputs as well as minimum quantum cost compare to existing design [15].

V. Performance Analysis

The realization of benchmark circuits analysis based on proposed algorithms by using programming language Java (jdk 1.7) on Netbeans IDE (8.0) in Window 7 Workstation is presented in Table II. All the experiment result are tested on Intel(R) Core(TM)i3 CPU@3.30GHz with 2GB RAM. Table II shows the experimental results for different benchmark functions and the comparison with the existing method [15] where the required number of gates, garbage outputs and ancilla inputs are minimized in notable manner. Finally, the trade-off between quantum cost and other factors summarizes the better optimization of proposed design of reversible PLAs is presented.

TABLE I
COMPARISON BETWEEN THE PROPOSED AND EXISTING [15] DESIGNS OF MULTI-OUTPUT FUNCTION F IN EQUATIONS (I)

RPLAs Design	Total Gates (GA)	Garbage (GB)	Ancilla Input(AI)	Quantum Cost(QC)
Proposed	13	5	7	37
Existing[15]	18	10	12	39

VI. SUMMARY AND FUTURE DIRECTIONS

In proposed design, the reusability of garbage outputs enhances zero energy dissipation which is the prime concern of reversible computing has been followed. This research has directed an novel approach to design reversible PLAs by proposing reversible PLAs grid and algorithms to construct AND and EX-OR planes. In the future, the success of reversible PLAs will raise the development of Reversible Field Programmable Logic Arrays [9].

REFERENCES

[1] R. Landauer, *Irreversibility and heat generation in the computing process*," *IBM J. of R&D*, vol. 5(3), pp. 183–191, 1961.

[2] C.H. Bennett, *Logical Reversibility of Computation*, *IBM J. of Research and Development*, vol. 17(6), pp. 525–532, 1973.

[3] H. Fleisher and L.I. Maissel, *An Introduction to Array Logic*, *IBM J. of Res. and Development*, vol. 19(2), pp. 98–109, 1975.

TABLE II
EXPERIMENTAL RESULTS USING DIFFERENT BENCHMARK FUNCTIONS

RPLAs	GA§(GA†)	GB§(GB†)	AI§(AI†)	QC§(QC†)
5xp1	166(132)	112(44)	115(47)	418(452)
9sym	427(377)	385(111)	377(103)	1405(1681)
adr3	67(52)	48(26)	46(24)	157(172)
apex3	3998(3719)	1799(520)	1795(516)	8654(9655)
b12	159(127)	132(56)	167(50)	453(495)
bw	305(281)	64(30)	87(53)	446(445)
cordic	12162(11389)	10640(1573)	10661(1552)	41694(50765)
duke2	941(874)	667(157)	674(164)	2735(3262)
e64	2170(2074)	2148(121)	2148(121)	8410(10282)
inc4	16(11)	10(4)	11(5)	34(35)
inc5	23(16)	16(6)	17(7)	53(56)
misex1	88(74)	51(25)	50(24)	193(206)
misex2	199(165)	176(79)	149(52)	122(673)
pdc	3801(3451)	3006(487)	3030(511)	12096(14115)
rd53	56(44)	42(23)	40(21)	134(141)
rd73	187(150)	141(68)	137(64)	487(550)
rd84	328(267)	265(114)	261(110)	928(1067)
sasao	14(9)	13(8)	10(5)	29(29)
sao2	284(257)	236(60)	230(54)	890(1065)
t481	49(40)	53(36)	38(21)	134(152)
table3	2602(2439)	1814(342)	1814(342)	7537(8995)
table5	2539(2384)	1819(319)	1682(182)	7516(8896)
xor5	8(4)	8(9)	4(5)	8(4)
z5xp1	167(139)	114(50)	117(53)	425(483)
z9sym	427(377)	385(111)	377(103)	1405(1681)

§Existing [15] (†Proposed) Reversible Programmable Logic Arrays)

[4] R.P. Feynman, *Quantum Mechanical Computers*, *Foundations of Physics*, vol. 16(6), pp. 507–531, 1986.

[5] E. Fredkin and T. Toffoli, *Conservative Logic*, *International Journal Theoretical Physics*, vol. 21, pp. 219–253, 1982.

[6] T. Toffoli, *Reversible Computing*, *Lecture Notes in Comp. Sci. on Auto., Lang. and Prog.*, vol. 85, pp. 632–644, 1980.

[7] B. Parhami, *Fault-tolerant Reversible Circuits*, *In proceding of 40th Asilomar conference on sig., sys. and comp., ACSSC'06, CA*, pp. 1726–1729, 2006.

[8] S.N. Mahammad and K. Veezhinathan, *Constructing Online Testable Circuits Using Reversible Logic*, *IEEE Trans. on Instrumentation and Measurement*, vol. 59(1), pp. 101–109, 2010.

[9] A.S.M. Sayem and S.K. Mitra, *Fault-tolerant Reversible Circuits*, *In proc. of IEEE Int'l conf. on computer sci. and automation engg., CSAE'11, Shanghai, China*, pp. 251–255, 2011.

[10] A.K. Biswas, M.M. Hasan, A.R. Chowdhury and H.M.H. Babu, *Efficient Approaches for designing Reversible Binary Coded Decimal Adders*, *Micro. J.*, vol. 39(12), pp. 1693–1703, 2008.

[11] M. Perkowski and et al, *A hierarchical approach to computer aided design of quantum circuits*, *In proc. of 6th Int'l sym. on R.&M. of Future Comp. Tech., Germany*, pp. 201–209, 2003.

[12] S.K. Mitra and A.R. Chowdhury, *Minimum Cost Fault Tolerant Adder Circuits in Reversible Logic Synthesis*, *In proc. of 25th Int'l conf. on VLSI Design, India*, pp. 334–339, 2012.

[13] H. Thapliyal and N. Ranganathan, *Design, synthesis and test of reversible circuits for emerging nanotechnologies*, *In proc. of IEEE comp. soc. annual sym. on VLSI, USA*, pp. 5–6, 2012.

[14] A.R. Chowdhury and R. Nazmul and H.M.H. Babu, *A new approach to synthesize multiple-output functions using reversible programmable logic arrays*, *In proc. of 19th Int'l conf. on VLSI Design (VLSID'06), India*, pp. 311–316, 2006.

[15] S.K. Mitra and L. Jamal and M. Kaneko and H.M.H. Babu, *An efficient approach for designing and minimizing reversible programmable logic arrays*, *In proc. of the Great Lakes sym. on VLSI, ACM, New York, NY, USA*, pp. 215–220, 2012.