

A New Algorithm for Reversible Logic Circuit Synthesis

Naga Kumar Gaju

Department of Electrical and Computer Engineering Texas Tech University,
Lubbock, Texas, USA

Abstract— In traditional CMOS technology the energy is expended in the form of loss of bits. This dissipation of energy is in the form of heat dissipation and plays a vital role in low power design of circuits. The conventional circuit design results in the Irreversible circuits which mean the number of outputs is not equal to the number of inputs which implies the loss of bits result in loss of energy. Circuit design based on reversible logic synthesis generate circuits with the number of outputs equal to the number of inputs resulting circuits with no energy loss and furthermore providing the advantage of retrieving the inputs from the outputs. The Reversible Logic design has application in extensive fields like Quantum Computing, Low power CMOS design, and Cryptography. The existing algorithms for realization of a Boolean function as reversible circuit is framed as a network of basic reversible gate library including Cnot gate, Toffoli gate, Fredkin gate etc known as Replacement based approach. In this paper a new algorithm for realizing a function as reversible circuit is based on Truth table approach is proposed and used to synthesize many benchmark circuits with simpler circuits, less gate count.

Keywords—*Reversible; bijective; Truth table approach; ancilla; gatecount*

I. INTRODUCTION

Energy loss is a vital consideration in any circuit design. The requirement for energy efficient and faster computing circuits leads to physical limitations. As predicted by Moore the transistor count in a chip will double everyone year. Shrinking in transistor size resulted in many implementational and operational difficulties like energy dissipation. Data processing is accompanied by a least amount of heat generation. According to Landauer's principle the loss in one bit of information will result in an estimated energy dissipation equal to $KT \ln 2$ Joules, where K is Boltzmann's constant and T is absolute temperature of operation [1]. In 1973 C.H. Bennett proved that this loss in energy and information can be conserved by making the computation reversible. Conventional computations by its nature are irreversible. Logically irreversible if the output does not uniquely define the inputs and the input cannot be retrieved from its output as all the input bits do not propagate till output[2]. Logical irreversibility implies physical irreversibility accompanied by dissipative effects. Due to limitations of conventional computing reversible computing seems to be the possible solution. Reversible computing saves energy dissipation by avoiding bit destruction.

Lauder's principle states that the source of heat generation

in computation is destruction of bits not their transformation[2]. The reversible computation is simply based on fact that, the existing information in any system can't just be destroyed but transformed in according to fact that at low level physics reversible means in closed system energy transforms from one state to another overtime in a mathematically invertible way. Because of less or no energy dissipation we can achieve high density and so we can achieve smaller size overcoming the physical limitations.

Any function of Boolean variables is said to be reversible if the number of outputs is equal to the number of inputs and function mapping from input vectors to output vectors is bijective function[4].

Any irreversible Boolean function can be made reversible by transforming the irreversible truth table to reversible truth table which requires extra inputs to bias it known as ancilla and extra outputs known as garbage outputs to hold the information which provides reversibility[5].

The important parameters to be considered in synthesis of reversible logic[4]: minimum number of ancilla, minimum number of garbage outputs, minimum number of gates, and minimum quantum cost.

II. PREVIOUS WORK

There are two approaches for conversion of irreversible circuit to reversible circuit:

1. Replacement based approach [Fig.1(a)]and
2. Truth table approach[Fig.1(b)].

The replacement based approach involves the irreversible circuit is directly converted into reversible circuits by replacement based reversible conversion.

The truth table is based on generating a truth table for the given circuit and then using reversible synthesis tools to generate reversible circuit.

The basic classifications with brief descriptions of methods

- 1.Composition method[4]: A Boolean function is realized as a network of small and well known reversible gates.
- 2.Decomposition method[4]: A Boolean function is decomposed into small functions which are realized as separate reversible networks.
- 3.EXOR logic based method[4]: Uses only Toffoli gate network to realize a Boolean function by decomposing it.
- 4.Search method[4]: A function is expanded and reduced with maintaining the output functionality unchanged. This method results in large circuits comparatively.

Schematic representation

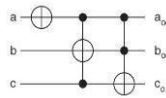


Fig.2.1(a)

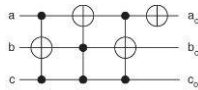


Fig.2.1(b)

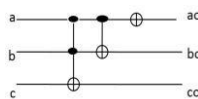


Fig.2.1(c)

Boolean function realization using existing algorithm[5]

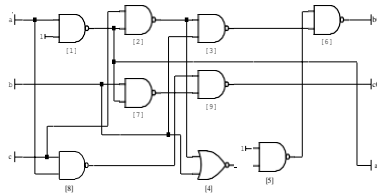


Fig.2.2(a)

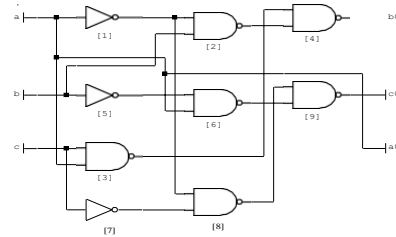


Fig.2.2(b)

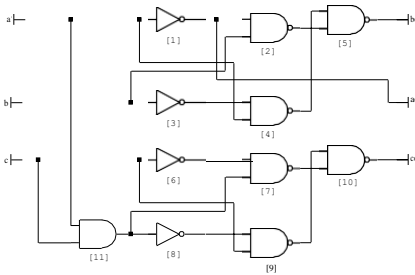


Fig.2.2(c)

Boolean function realization using proposed algorithm

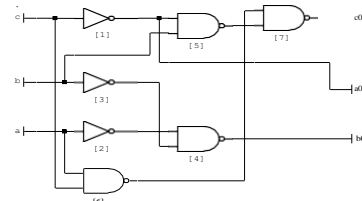


Fig.2.3(a)

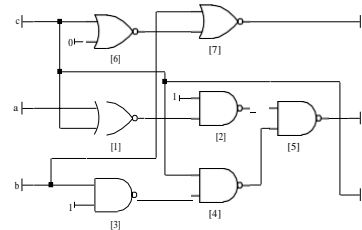


Fig.2.3(b)

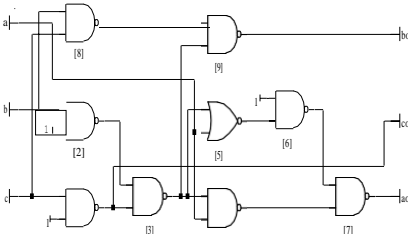


Fig.2.3(c)

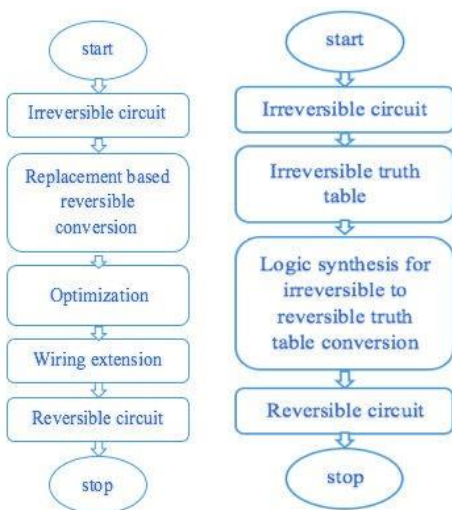


Fig.1(a) Replacement based Fig.1 (b)Truth table based approach

In this paper the truth table-based approach is used for reversible logic synthesis. The circuits generated for the functions with schematic representation Fig.2.1(a), Fig.2.1(b) and Fig.2.1(c) using replacement based approach are Fig.2.2(a), Fig.2.2(b) and Fig.2.2(c) and the truth table based approach are Fig.2.3(a), Fig.2.3(b) and Fig.2.3(c) respectively showing the gate count. The algorithm proposed in this paper resulted in circuits with less number of gates compared to existing algorithms.

III. DEFINITIONS

A. Irreversible Logic Function

A Boolean function is said to be irreversible if the outputs of the function does not uniquely define the inputs.

B. Bijective Function

A function is bijective if each element in the input set has a unique mapped output also known as one-to-one mapping.

	F	G1	G2		F	G1	G2		F	G1	G2		F	G1	G2		F	G1	G2
0	1			1	0			1	0			1	0	0	0	0	1	0	0
1	0			0	1			0	1			0	1	0	0	1	0	0	0
2	0			2	0			2	0			2	0	1	0	2	0	1	0
3	1			3	1			3	1			3	1	1	0	3	1	1	0
4	0			4	0			4	0			4	0	0	1	4	0	0	1
5	1			5	1			5	1			5	1	0	1	5	1	0	1
6	1			6	1			7	0			7	0	1	1	6	1	1	1
7	0			7	0			6	1			6	1	1	1	7	0	1	1
Step:1				Step:2				Step:3				Step:4				Step:5			

Fig. 3 An example illustrating step by step procedure for multiple input and single output algorithm

C. Reversible Logic Function

A Boolean function is said to be reversible if the number of the outputs and the inputs are equal and the outputs of the function have the unique preimage. If the function is one-to-one mapping or bijective function.

D. Ancilla

The constant input added to the circuit to make it reversible and whose original state is known in advance.

E. Garbage outputs

Refers to the number of outputs added to make a circuit reversible such that

$$\text{Number of inputs} + \text{Ancilla} = \text{Number of outputs} + \text{Garbage outputs}$$

F. Gate count

The number of logic gates used in realization of reversible logic circuit.

IV. ALGORITHM FOR MULTI-INPUT SINGLE OUTPUT

A. Algorithm

Let $f(a_1, a_2, a_3, \dots, a_n) = \sum (m_0, m_1, \dots, m_m)$ be a Boolean function defined in terms of minterms and let the number of minterms be m.

Classifying the given functions into two cases:

- 1) with number of minterms(m) equal to half of the number of input vectors $2^n/2$.
- 2) with number of minterms(m) greater than or less than the number of input vectors.

Case I: If number of minterms = half of the number of input vectors

- 1) Fill the truth table marking the corresponding minterms
- 2) number the sequence of 0's and 1's from 0 to $2^n - 1$
- 3) arrange the sequence in the pattern of Least Significant Bit(LSB) or the Most Significant Bit(MSB)
- 4) add the garbage outputs by filling the missing components of the corresponding vector with 0's and

1's such that a unique vector exists, and number of inputs are equal to number of outputs

- 5) rearrange back the sequence to the initial sequence
- 6) using Quine McCluskey technique represent the output functions in minimized form
- 7) Choose the garbage output functions which uses minimum number of gates to realize corresponding to the arranged sequence in the pattern of MSB or LSB.

Case II: If number of minterms not equal to half the number of input vectors

Constant input ancilla is added to make the function reversible
 > the outputs of the reversible circuit include the original inputs and ancilla $\oplus f(a_1, a_2, \dots, a_n)$

B. Example

Case I: Let $f(a_1, a_2, a_3) = \sum (0, 3, 5, 6)$ be the given Boolean function. To make it a reversible function the number of outputs must be equal to number of inputs (n=3) we must add two garbage outputs.

For the given function the number of minterms (m) = 4, implies $m=2^{3-1}$ belongs to the first category no ancilla required.

Minimizing the columns in the Step v using the Quine McCluskey method results in the best possible minimized functions correspondingly are:

$$F: \overline{a_1} \overline{a_2} \overline{a_3} + \overline{a_1} a_2 a_3 + a_1 \overline{a_2} a_3 + a_1 a_2 \overline{a_3} \quad (1)$$

$$G1: a_2 \quad (2)$$

$$G2: a_3 \quad (3)$$

F is the main function and G1, G2 are the garbage outputs

Case II: Let $f(a_1, a_2, a_3) = (0, 3)$ be the given Boolean function. The number of minterms in the given function are not equal to half the number of input vectors. This falls into second category. Ancilla(x) is added to the input and the outputs are a_1, a_2, a_3 and $x \oplus f(a_1, a_2, a_3)$

$$\text{Inputs: } a_1, a_2, a_3, x \quad (4)$$

$$\text{Outputs: } a_1, a_2, a_3, \text{ and } \overline{a_1} \overline{a_2} \overline{a_3} \overline{x} + a_1 a_2 \overline{a_3} \overline{x} + \overline{a_1} a_2 x + a_3 x \quad (5)$$

Algorithm I: For single output Boolean function

Input: number of inputs(n), number of minterms(m) and minterms
Output: reversible logic with main function and garbage outputs in minimized form

```

begin
  Read n, m, minterms
  If m=2^n-1
    for i=0 to 2^n-1
      do if output function has minterm i
        a[0][i]=1
      else
        a[0][i]=0
      end
      arr[i]=i
    end
    for i=0 to 2^n-1
      do
        count=0
        if a[0][i%2] Th 0
          for j=1 to 2^n-1
            do
              if a[0][j]=1 & count=0
                swap a[0][i] a[0][j]
                swap arr[i] & arr[j]
                count=count+1
              end if
            end if
          end if
        end
        for k=1 to n-1
          do
            for i=0 to 2^n-1
              fill a[k][i] as 2^k 0's 2^k 1's
            end
          end
        end
      do
        constant input ancilla is added to make the
        function reversible
        the outputs of the reversible circuit include the
        original inputs and ancilla ⊕ f(a1, a2,
        ..., an)
      end
    end
  end
  Using Quine McCluskey technique
  finding prime implicants
  choosing essential prime implicant
  Print the output functions in minimized form
end
end
  
```

	F1	F2	G	G	G	G	G	G	G	G	
0	0	0		0	0	0	0	0	0	0	
1	1	0		0	0	0	0	0	0	0	
2	0	1		0	0	0	0	0	0	0	
3	1	1		0	0	0	0	0	0	0	
4	0	0		0	0	0	1	1	1	1	
5	1	0		0	0	0	0	1	1	1	
6	0	1		0	0	0	0	0	1	1	
7	1	1		0	0	0	0	0	0	1	
	Step:1			2	3	4	5	6	7	8	9

Fig. 4 An example illustrating step by step procedure for proposed algorithm for multiple output

V. ALGORITHM FOR MULTI-INPUT MULTI OUTPUT

A. Algorithm

Let $f1(a1, a2, a3, \dots, an) = \sum(m0, m1, \dots, mm1)$, $f2(a1, a2, a3, \dots, an) = \sum(m0, m1, \dots, mm2), \dots, \dots, \dots$, $fout(a1, a2, a3, \dots, an) = \sum(m0, m1, \dots, mmout)$ be a Boolean functions defined in terms of minterms and let the number of minterms be $m1, m2, \dots, mout$ in each output function respectively.

Classifying the given functions into two cases:

Case I: If number of 1's in $f1 \oplus f2 \oplus \dots \oplus fout = m1 = m2 = \dots = mout$

If the row of outputs occurs for the first time join '0' to the row

else join '1' to the row Continue until number of outputs equal number of inputs

Case II: Else

a constant input ancilla is added to make the function reversible Outputs are ancilla $\oplus f1$, ancilla $\oplus f2, \dots, \dots, \dots, \text{ancilla} \oplus fout$ and the functions as a result of repeating case I after adding ancilla

B. Example

Case I: Let number of inputs $n=3(a1, a2, a3)$, number of outputs $out=2$, number of minterms in first output function $m1=4$, number of minterms in second output function $m2=4$
 $f1(a1, a2, a3) = \sum(1, 3, 5, 7)$ and
 $f2(a1, a2, a3) = \sum(2, 3, 6, 7)$

For given functions number of ones in $f1 \oplus f2 = m1 = m2$

Minimizing the columns in the Step v using the Quine McCluskey method results in the best possible minimized functions correspondingly as

$$F1: a1 \tag{6}$$

$$F2: a2 \tag{7}$$

$$G: a3 \tag{8}$$

F1, F2: main function

G: Garbage functions

Case II: Let $n=3$, $out=2$, $m1=3$, $m2=4$, $f1(a1,a2,a3) = \sum(1,3,5)$ and $f2(a1,a2,a3)=\sum(1,3,5,7)$
Ancilla(x) is added to the input and the outputs are

$$F1: \quad \overline{a2a3}x + \overline{a1a3}x \quad (9)$$

$$F2: \quad xa3 + \bar{x}a3 \quad (10)$$

$$G1: \quad a1a2a3 \quad (11)$$

$$G2: \quad xa1a2a3 \quad (12)$$

F1, F2: Main functions G1, G2: Garbage functions

Algorithm I: For multiple output Boolean function

Input: Number of input variables(n), number of minterms in each function(mi), minterms in the function

Output: Main function and garbage outputs in minimal form
begin

Read n , number of outputs(out), number of minterms in each function(array $arrm$), minterms in each output function (array $arrf1, arrf2, \dots, arrf_{out}$)

If

for $i=0$ to $out-1$

if number of ones in

$arrf1 \oplus arrf2 \oplus \dots \oplus arrf_{out} = arrm[i]$

end if

for $i=out$ to $n-1$

do for $j=0$ to 2^n-1

do for $k=0$ to $out-1$

do

if j^{th} row occurs for first time

$arrf_{out+1}[j][i]=0$

else j^{th} row is repeated

$arrf_{out+1}[j][i]=1$

end if

end

end

end

else

do

constant input ancilla is added

outputs are ancilla $\oplus arrf1$,

ancilla $\oplus arrf2 \dots$ ancilla $\oplus arrf_{out}$ and functions resulted by running case I after adding ancilla

end

end

do

Using Quine McCluskey technique

finding prime implicants

choosing essential prime implicant

Print the output functions in minimized form

end

end

VI. RESULTS

The proposed algorithm is used to synthesis various Boolean functions and the results with comparison in terms of gate count are as follows:

TABLE I
GATE COUNT COMPARISON FOR SOME STANDARD FUNCTIONS FROM[8]

Function		Using Fredkin gate		Using Peres gate		Using Toffoli gate		Using Proposed algorithm	
		AND	OR	AND	OR	AND	OR	AND	OR
<i>Gate Count</i>									
F1	AB'C	8	4	8	4	9	6	4	3
F2	AB	4	2	4	2	3	2	3	2
F3	A'BC+A'B'C'	12	6	16	8	12	6	5	4
F4	A'BC+AB'C'	16	8	12	6	-	-	5	4
F5	A'B+BC'	20	10	20	10	12	8	4	3
F6	AB'+A'BC	12	6	32	16	-	-	4	2
F7	A'BC+ABC'+A'B'C'	20	10	20	10	-	-	7	3
F8	AB+BC+CA	32	16	4	22	-	-	7	6
F9	A'B+B'C	4	2	16	8	12	8	6	4
F10	ABC'+A'B'C'+AB'C+A'BC	44	22	16	8	6	4	4	3

TABLE II
GATE COUNT COMPARISON FOR BENCHMARK CIRCUITS FROM[9]

Benchmark Circuit		Ancilla		Gate count (existing)		Gate count (proposed)	
		existing	proposed	AND gates	OR gates	AND gates	OR gates
1	2of5	2	0	20x3	20x2	12	11
2	rd32	1	0	4x3	4x2	7	9
3	4_49	0	0	13x3	13x2	16	12
4	xor5	0	0	4x3	4x2	14	12
5	4mod5	1	1	5x3	5x2	8	7
6	5mod5	1	1	11x3	11x2	14	13
7	ham3	0	0	5x3	5x2	9	6
8	hwb4	0	0	15x3	15x2	16	12
9	6sym	0	0	14x3	14x2	26	25
10	9sym	0	0	73x3	73x2	74	73
11	alu	0	1	18x3	18x2	29	19
12	majority3	0	0	4x3	4x2	9	6
13	majority5	0	0	16x3	16x2	35	30
14	5one013	0	0	19x3	19x2	42	36
15	5one245	0	0	20x3	20x2	45	30
16	5one013	0	0	19x3	19x2	42	36
17	5one245	0	0	20x3	20x2	45	30
18	4b15g_1	0	0	15x3	15x2	23	19
19	4b15g_2	0	0	15x3	15x2	34	29
20	4b15g_3	0	0	15x3	15x2	29	24
21	4b15g_4	0	0	15x3	15x2	31	27
22	nth prime3	0	0	4x3	4x2	8	5
23	nth prime4	0	0	12x3	12x2	23	16
24	nth prime5	0	0	25x3	25x2	39	34
25	Mod5adder	0	0	19x3	19x2	25	19

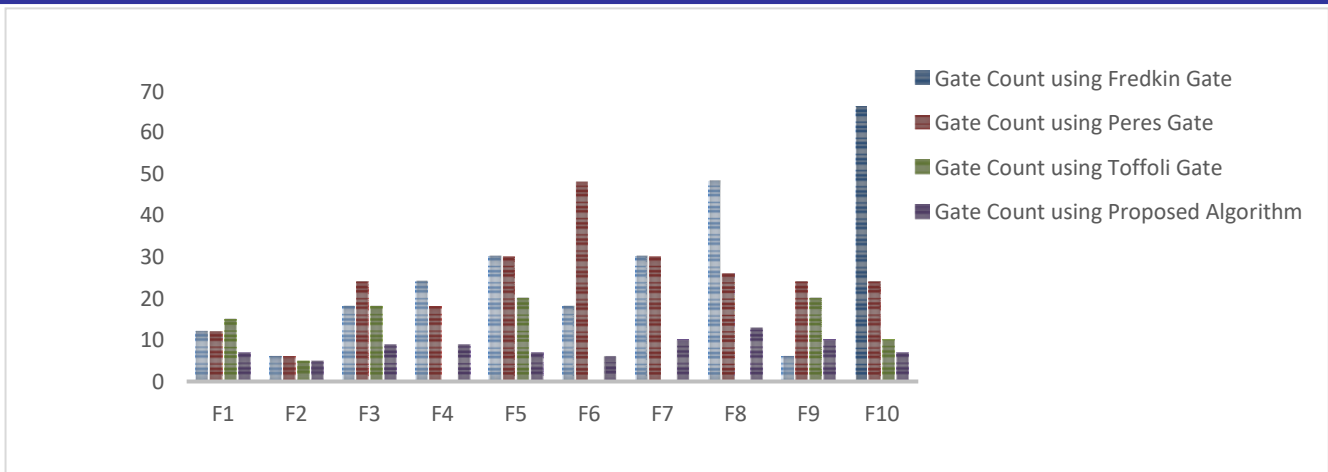


Fig. 5 Graph showing Gate Count for standard function Table I

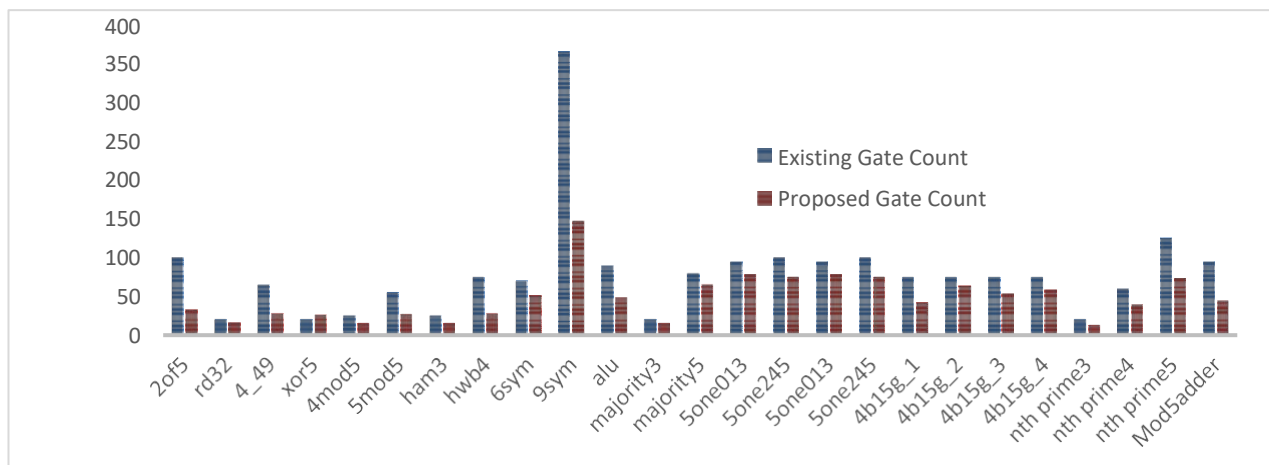


Fig. 6 Graph showing Gate Count for standard function Table II

VII. CONCLUSIONS

An algorithm and a tool are described that uses truth table approach to synthesize the reversible logic circuits is framed. The algorithm uses the back tracing and bijective mapping technique to fill the truth table to find the garbage functions and ancilla if any in the minimized form using the Quine-McCluskey minimization within 2^{n-1} steps for n number of inputs. The tool designed basing on the proposed algorithm is used the synthesize the functions tabulated in table I and table II showing the reduced gate count of 36 percentage on average to realize the functions compared to existing algorithm. The proposed algorithm efficiently synthesizes the reversible function without using any network basic reversible gates such as Toffoli or Fredkin resulting in the reduced gate count.

VIII. REFERENCES

- [1] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J. Res. Develop.*, vol. 5, no. 3, pp. 183–191, July 1961.
- [2] Micheal P. Frank, "The Physical Limits of Computing", Vol.4, No.3, pp. 16-26, May-June 2002, doi:10.1109/5992.998637, IEEE, 2002.
- [3] C.H. Bennett, "Logical Reversibility of Computation", IBM Research and Development, pp. 525-532, November 1973
- [4] "Garbage in Reversible Designs of Multiple Output Functions", Dmitri Maslov and Gerhard W. Dueck, Faculty of Computer Science, University of New Brunswick
- [5] "Irreversibility and Heat Generation in the Computing Process" IBM Journal of Research and Development (Volume: 5, Issue: 3, July 1961)
- [6] "An Algorithm for Synthesis of Reversible Logic Circuits", Pallav Gupta, *Student Member, IEEE*, Abhinav Agrawal, and Niraj K. Jha, *Fellow, IEEE*, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 25, NO. 11, NOVEMBER 2006
- [7] "New Universal Gate Library for Synthesizing Reversible Logic Circuit Using Genetic Programming", Mustapha Yusuf Abubakar, Low Tang Jung, Mohamed Nordin Zakaria, Ahmed Younesy and Abdel-Haleem Abdel-Atyz, 2016 3rd International Conference On Computer And Information Sciences (ICCOINS)
- [8] "Basic Logic Gate Realization using Quantum Dot Cellular Automata based Reversible Universal Gate", Saroj Kumar Chandra, Prince Kumar Sahu, International Journal of Computer Applications (0975-8887)
- [9] Reversible Benchmark Circuits. Available: <http://webhome.cs.uvic.ca/~dmaslov/>