Asian Journal of Information Technology 4 (12): 1146-1151, 2005

Minimization of Reversible Adder Circuits

[1]Saiful Islam and [2]Rafiqul Islam
[1]Deptartment of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh
[2]School of Information Technology, Deakin University, Melbourne, Australia

Abstract: Losing information causes losing power. Information is lost when the input vector cannot be uniquely recovered from the output vector of a combinational circuit. The input vector of reversible circuit can be uniquely recovered from the output vector. In this study we have emphasized on the design of reversible adder circuits that is efficient in terms of gate count, garbage outputs and quantum cost and that can be technologically mapped. It has been analyzed and demonstrated that the results of our proposed adder circuits shows better performance compared to similar type of existing designs. Technology independent equations required to evaluate these circuits have also been given.

Key words: Reversible logic, garbages, quantum costs, full adder, carry skip logic

INTRODUCTION

Power dissipation is a very important factor in VLSI design. The main benefit of reversible logic is theoretically zero power dissipation. The input vector of reversible circuit can be uniquely recovered from the output vector, that is, for each input pattern there is a unique output pattern. It naturally takes care of heating generated due to the information loss. Landauer[1-2] proved that logic computations that are not reversible necessarily generate heat irrespective of their implementation technologies. According to[3] zero energy dissipation would be possible only if the network consists of reversible gates. Thus reversibility will become an essential property in future circuit design. For reversible logic history refer to[4].

Synthesis of reversible circuits differs significantly from the synthesis of combinational or classical logic circuits. Because in a reversible circuit the number of inputs must be equal to the number of outputs, no fan-out is permitted and the resulting circuit must be acyclic. Therefore, a good synthesis method must take into account the following rules:

- Use as many outputs of every gate as possible and thus minimize garbage outputs.
- Do not create more constant inputs to gates than is absolutely necessary.
- Avoid leading output signals of gates to more than one input, because each fan-out of two requires adding one copying circuit.

Adders are fundamental building blocks in many computational units. The anticipated paradigm shift logic compatible with optical and quantum computing requires compatible adder implementations. Reversible adder circuits and their minimization issue have been described in[5-10]. In this study we have design reversible adder circuits that are efficient in terms of gate count, constant input and garbage outputs. We have also computed the quantum cost of our proposed adder circuits and others and found that the quantum cost of our design is low. Technology independent evaluations of our adder circuits have also been given.

Definition: A gate or a circuit is called reversible if there is a one-to-one correspondence between its input and output assignments. Any reversible circuit realizes only the function that is reversible.

Definition: The function f $(x_1, x_2, ...x_n)$ of n boolean variables is called reversible if

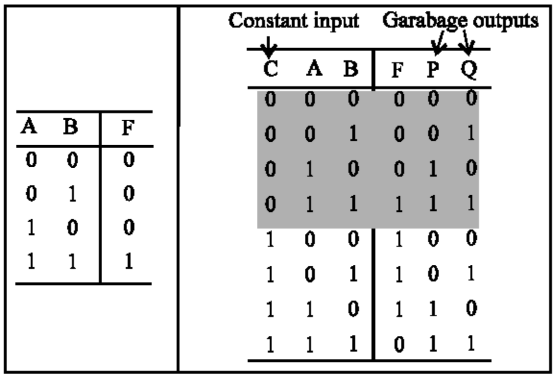
- The number of outputs is equal to the number of inputs.
- Each input pattern maps to a unique output pattern.

In other words, reversible functions perform permutations of the set of input vectors.

Definition: Garbage is the number of outputs added to make an n-input k-output function, i.e., (n, k) function, reversible.

The word constant-input denotes the inputs that were added to an (n, k) function to make it reversible. The relation between garbage outputs and constant inputs can be shown by the following simple formula (Fig. 1).

Corresponding Author: Saiful Islam, Deptartment of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh


Constant input
Garbage outputs

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

C	A	B	F	P	Q
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

Fig. 1: Garbage and constant input required to make an (n, k) function reversible

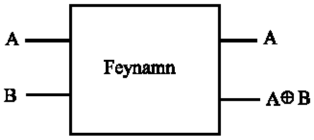


Fig. 2: 2*2 Feynman gate

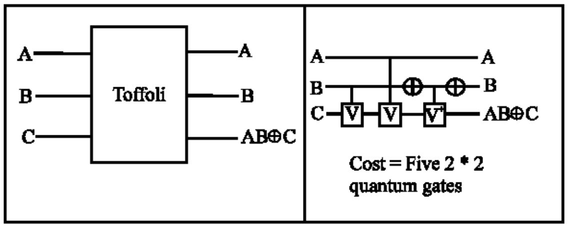


Fig. 3: 3*3 Toffoli gate

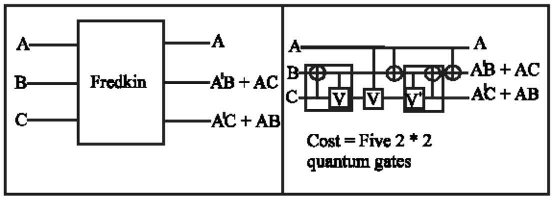


Fig. 4: Fredkin gate

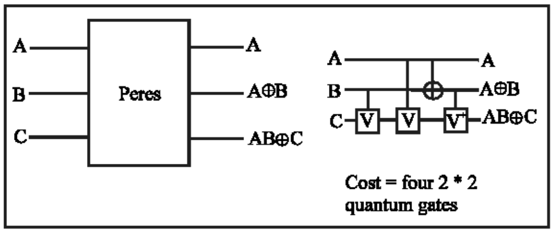


Fig. 5: Peres gate

Input + constant input = output + garbage

Families of reversible gates and their quantum cost: There exist many universal reversible gates[11-15]. There exists only one 1*1 reversible gate called inverter ($A \rightarrow \bar{A}$). This gate is very important since it does not introduce garbage outputs. Some of the popular and important gates and their quantum cost are described below.

Feynman gate: A 2*2 Feynman gate, also called CNOT, output is given by the equation $(A, B) \rightarrow (P = A, Q = A \oplus B)$. This gate is one through because it passes one of its inputs. Every linear reversible function can be built by using only 2*2 Feynman gates and inverters (Fig. 2).

The detailed cost of a reversible gate depends on any particular realization technology of quantum logic. According to Perkowski *et al.*[16], it is assumed that the cost of every 2*2 is the same. A 1*1 cost nothing, since it can be always included to arbitrary 2*2 gate that precedes or follows it. Thus, every permutation quantum gate will be build from 1*1 and 2*2 quantum primitives and its cost calculated as a total sum of 2*2 gates.

Toffoli gate: A 3*3 Toffoli gate, also called controlled controlled-NOT (CCNOT), output is given by the equation $(A, B, C) \rightarrow (P = A, Q = B, R = AB\oplus C)$. This gate is two through because two of its outputs are identical of its inputs. Using the well known realization of Toffoli gate with truly quantum 2*2 primitives, as shown in Fig. 3 according to Perkowski *et al.*[16], the cost of Toffoli gate is five 2*2 gates, or simply, 5. In Fig. 3 V is a square-root-of-NOT gate (unitary matrix V) and V^+ is its hermitian. Thus VV^+ creates a unitary matrix of NOT gate and $VV^+ = 1$ (an identity matrix, describing just a quantum wire).

Fredkin gate: A 3*3 Fredkin gate output is given by the equation $(A, B) \rightarrow (P = A, Q = A'B + AC, R = A'C + AB)$. This gate is conservative, that is, the Hamming weight (number of logical ones) of its input equals the Hamming weight of its output. The Fredkin gate is sometimes called the controlled permutation gate (Fig. 4).

The cost of Fredkin gate is exactly the same as the cost of Toffoli gate[15] and not more than of it, as some authors assumed and as may be suggested by classical binary schema of such gates, where the Toffoli gate includes a single Davio gate, while the Fredkin gate includes two multiplexers.

Peres gate: A new permutation quantum gate[11] with equation: $(A, B) \rightarrow (P = A, Q = A \oplus b, R = AB \oplus C)$ can be realized with cost 4 (Fig. 5). It is just like a Toffoli gate from Fig. 3 but without the last Feynman gate from right. This is the cheapest quantum realization of a complete (universal) permutation gate[16].

Composition of full adder circuit

Theorem: A full-adder can be realized with at least two garbage output and one constant input.

Proof: The full-adder output S = (A ⊕ B ⊕ Ci_n), C_{out} = ((A ⊕ B) Ci_n ⊕ AB) equations produce the same output (1, 0) for the three distinct input combinations (0, 0, 1), (0, 1, 0) and (1, 0, 0). Therefore, to separate all repeated values of outputs S and C_{out} we need at least two garbage outputs. Thus, a total of outputs is 2+2 = 4. Since in a reversible circuits number of inputs must be equal to number of outputs and there are three inputs A, B and C_{in} at least one constant input is necessary.

A full adder implementation using two 3*3 Toffoli gates and two 2*2 Feynman gate is presented in[5]. The circuit requires 4 reversible gates, produces 2 garbage outputs and the quantum cost is of 10. Another full adder implementation using four 3*3 Fredkin gates is presented in[9]. The circuit requires four reversible gates, produces two garbage outputs and the quantum cost is of 20. The other full adder implementation has found in Azad Khan[6] and Hasan *et al.*[8]. The full adder implementation in[6] using 3*3 Khan Gate and 2*2 Feynman gate requires 3 reversible gats and produces 3 garbage outputs. The full adder circuit described by Hasan *et al.*[8] requires 3 reversible gates and produces 2 garbage outputs. We have proposed a full adder circuit implementation shown in Fig. 6 that requires only two reversible gates, produces 2 garbage outputs and 1 constant input (Theoretical minimum) and quantum cost is of only 8. The quantum cost of the full adder circuits in Azad Khan[6] and Hasan *et al.*[8] cannot be calculated since the reversible gate they used, quantum implementation of that is not available yet. To evaluate our proposed full adder circuit we have also calculated other costs namely unit clock cycles requirement, EX-OR, two input AND and NOT gate calculations. We have shown that our full adder circuit is also efficient in terms of these costs. These are summarized in Table 1.

Table 1: Gate count, Garbage outputs, Constant inputs, Quantum cost, unit Clock Cycle, EX-OR, AND and NOT gate calculations. The dash (-) in the column indicates that the quantum cost of these adder circuits cannot be computed because of Khan gate for which there is no quantum implementation

Full-adder composition	No. of gates used	No. of garabage output	No. of instant input	Quantum cost	Unit clock cycle	Two inputEXOR gate calculations	Two input AND gate calculations	NOT gate calculations
Proposed Peres	2	2	1	8	2	4	2	0
Toffoli, Khan and Feynman[8]	3	2	1	-	3	4	3	0
Toffoli and Feynman[5]	4	2	1	10	4	4	2	0
Khan and Feynman gate[6]	3	3	2	-	3	5	4	6
Fredkin[9]	4	3	2	20	4	8	16	4

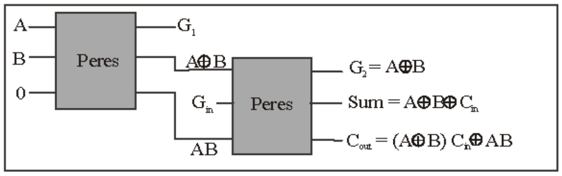


Fig. 6: Peres Full Adder (PFA); requires two 3*3 Peres gate, produces 2 garbage outputs, requires1 constant input and quantum cost is of only 8

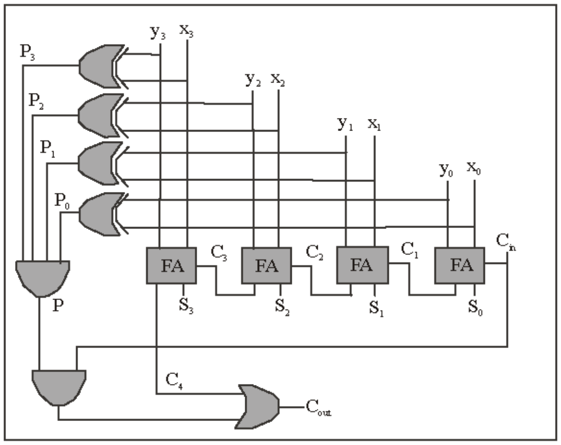


Fig. 7: 4-Bit carry skip adder

Carry skip adder: The carry skip adder reduces the delay due to the carry computation. Consider the full-adder's operation. If either input is a logical 1, the cell will propagate the carry input to the carry output. Therefore, the ith full-adder carry input, C_{in}, will propagate the carry input to its carry output, C_i + 1, when P_i = A_i ⊕ B_i. Furthermore, multiple full-adders, called a block, can generate a block propagate signal to detour the incoming carry around to the block's carry output signal. Figure 7 shows a 4 bit carry skip adder block. Each block is a small ripple carry adder producing the block's sum and carry bits. However, each block quickly calculates whether the block's carry input is propagated to its carry output.

Carry skip adder block using peres full adder: The carry skip adder propagate the block carry input to the next block if block group propagate signal P is one. The conventional skip block in Fig. 7 uses the AND-OR gate combination. A carry skip block based on Peres gates can be constructed by replacing the AND and OR gates with their respective Peres gate implementation.

The traditional carry skip AND-OR logic in Fig. 7 and the carry skip logic in Fig. 8 do not process equivalent truth tables. The AND-OR carry skip generates a block carry out, i.e., $C_{out} = 1$, when the most significant full adder carry C_4 equals one, the block propagate P equal one and the block carry input C_{in} signal equals zero. The proposed carry skip logic generates block carry out $C_{out} = 0$ under the same inputs. Either result is appropriate, because C_4 cannot equal one when $C_{in} = 0$ and $P = 1$. However, it must be noted that the Peres carry skip logic more faithfully adheres to the sprit of carry skip addition by propagating the correct value of C_{in} to C_{out}. Thereby, carry propagation performance is improved in block inputs that are possible.

The Peres carry skip logic in Fig. 8 can improve carry propagation if the block carry propagate signal P is one. When $P = 1$, the block carry input C_{in} should propagate to the next block, regardless of the result of the carry C_4 created by the full adders within the block. Because the traditional AND-OR carry logic in Fig. 7 must account for sending C_4 when $P = 0$, it does not perform its carry skip operation efficiently. Consider the carry skip adder block in Fig. 7 when $C_{in} = 1$, $P = 1$, $C_4 = 1$. Obviously, the block's carry out C_{out} is one. Now consider when the same block performing its next operation with C_{in} and $P = 1$. Therefore, the full adders must generate $C_4 = 0$. The carry ripple through the full adders will take some time. Meanwhile, the AND-OR logic will postpone generating theC_{out} until C_4 is resolved. The Peres carry skip logic in Fig. 8 passes C_{in} to C_{out} whenever $P = 1$, regardless of C_4. The timesavings can be significant as block size increases.

A B bit full adder requires 2B Peres gate using the circuit in Fig. 6. The B input AND gate requires B − 1 Peres gates. Therefore, a B bit carry skip adder requires and 3B Peres gate using the circuit in Fig. 6.

Consider the B bit carry skip adder block in Fig. 8 generating a block carry out C_{out}generates via carry ripple through the full adders. The least significant full adder requires a path delay of 2 Peres gates to generate C_1 from the addends. Then, the carry ripples through the subsequent full adders with a path delay of 1 Peres gate per bit. Finally, the Peres gate in the left of Fig. 8 generates C_{out}. Therefore, the delay to generate block carry out C_{out} (via ripple) with a C_{out} bit carry skip adder is

$$d_{ripple}(B+1) = B+1 \tag{1}$$

Consider the B bit carry skip adder Block in Fig. 8 generating a block carry out C_{out} generates via a carry skip. Each full adder requires a path delay of 2 Peres gate to generate P_1 from the addends (all full adders generate P_1 simultaneously). Then, all propagate signals for the carry skip adder block are combined with a B bit AND gate with delay $|\log_2^N|$. Finally, the Peres gate in the left of Fig. 8 generates C_{out}.

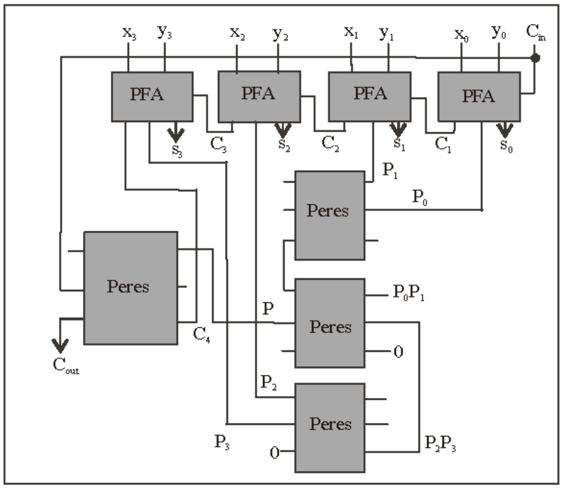


Fig. 8: 4-Bit carry skip adder block using peres full adder

The full adder in Fig. 6 generate sum bit S_1, carry bit C_1 and propagate signal P_1 (G_2) passing through 2, 2 and 2 reversible gate. Therefore, the delay to generate S_1 is 2. The delay to generate P_1 is 2. And the delay to generate C_1 is 2 reversible gates. Then, all propagate signals for the carry skip adder block are combined with a B bit AND gate with delay $|\log_2^N|$. Finally, the Peres gate in the left of Fig. 8 generates C_{out}.

$$d_{skip} = \lceil \log_2^B \rceil + 3 \tag{2}$$

The total (worst case) delay T_{fixed} of an N bit carry skip adder with fixed block size B is the sum of the ripple carry delay through the first carry skip adder block, skip delays through the intermediate blocks and the ripple carry delay through the last block, or

$$T_{fixed} = (B+1) + \left(\frac{N}{B} - 2\right)\left(\lceil \log_2^B \rceil + 3\right) + (B+1) \tag{3}$$

Assuming $\lceil \log^B{}_2 \rceil \approx B/2$ and minimizing T_{fixed} with respect to block size B gives

$$T_{fixed} = B + \frac{N}{2} + \frac{3N}{B} - 4 \tag{4}$$

Minimizing T_{fixed} with respect to block size B gives

$$1 - \frac{3N}{B^2} = 0 \text{ or } B_{opt} = 1.73\sqrt{N} \tag{5}$$

Substituting (5) into (4) gives the shortest delay for a fixed block size carry skip adder

$$T_{\text{fixed}} = \frac{N}{2} + 3.47\sqrt{N} - 4 \qquad (6)$$

Variable block carry skip logic: Varying the size of the carry skip adder blocks can reduce the total worst-case delay. Since carries generated or absorbed in the adder circuits have shorter data paths. Without loss of generality, the first and last carry skip blocks are b bits wide and the carry skip adder is divided into t blocks, where t is even. Assuming the carry skip adder block sizes are

$$b \; b+1 \ldots b+\frac{t}{2}-1 \;\; b+\frac{t}{2}-1 \ldots b+1 \; b \qquad (7)$$

Summing the number of bits in the blocks, equating to N and rearranging gives

$$b = \frac{N}{2} - \frac{t}{4} + \frac{1}{2} \qquad (8)$$

The total worst case delay T_{variable} of an N bit carry skip adder with the variable block sizes is the sum of the ripple carry delay through the first carry skip adder block, the skip delays through the intermediate blocks and the ripple carry delay through the last block. Assuming the variable block sizes in[5]s, the total delay is

$$T_{\text{var iable}} = 2(b+1) + 2\sum_{k=b+1}^{b+\frac{t}{2}-1}\left(\left\lceil \log_2^k \right\rceil + 3\right) \qquad (9)$$

Assuming $\lceil \log_2{}^{k/2} \rceil$ and rearranging[9] becomes

$$T_{\text{var iable}} = 4b - 4 + 3t + \frac{\left(b+\frac{t}{2}-1\right)\left(b+\frac{t}{2}\right) - b\left(b+1\right)}{2}$$

$$= \frac{t^2}{8} + \frac{11t}{4} + \frac{bt}{2} + 3b - 4 \qquad (10)$$

Inserting (8) into (10) and collecting terms gives

$$T_{\text{var iable}} = \frac{9t}{4} - \frac{5}{2} + \frac{3N}{t} + \frac{N}{2} \qquad (11)$$

The optimal number of blocks is found with

$$\frac{\partial T_{\text{var iable}}}{\partial t} = \frac{9}{4} - \frac{3N}{t^2} = 0 \text{ or } t_{\text{opt}} = \frac{2}{3}\sqrt{3}\sqrt{N} \qquad (12)$$

Therefore, the optimal variable block size carry skip adder has delay

$$T_{\text{var iable}} = \frac{N}{2} + \sqrt{3}\sqrt{N} - \frac{5}{2} \qquad (13)$$

RESULTS AND DISCUSSION

We compare our proposed full adder circuits with existing designs and result is shown in Table 1. In the previous Quantum costs of those circuits are not considered. We calculate the Quantum cost of those adder circuits and compare them with our proposed designs.

Table 2: Showing T_{fixed} for different implementations

Block size in No. of bits	T_{fixed} for [9]	T_{fixed} for Peres
4	13.49	4.93
8	21.9	9.80
16	34.98	17.86
32	55.82	31.60
64	89.97	55.71
128	147.64	99.19
256	247.94	179.43
512	427.27	330.38
1024	7555.87	618.85
2048	1370.54	1176.77
4096	2539.74	2265.70

The analytical performance of the carry skip adder of Brace *et al.*[9] and our carry skip adder (Fig. 8) is given in Table 2. It is evident from Table 2 that our design performs better. For smaller block size our carry skip adder performs best (approximately double) and practically smaller block size is required. We choose binary exponential values for the block size, which is natural for block size.

CONCLUSIONS

In this study, we have presented the reversible synthesis of quantum cost efficient full adder circuit and using this adder proposed the design of other adder circuits for which it is used as a fundamental building block. The main goal of our paper is finding a good architecture for adder circuits using reversible logic based on minimizing gate count, garbage outputs, constant inputs and quantum cost. We have compared our proposed adder circuits with the previous one after computing their quantum cost based on current quantum technology (i.e., NMR[16]). Technology independent analysis of our proposed adder circuits is also given since quantum or optical logic implementations are not mature at this stage (i.e., Khan gate[6] for which there is no quantum implementation of this gate is available).

REFERENCES

1. Landauer, R., 1961. Irreversibility and heat generation in the computing process. IBM J. Res. Develop., 5: 183-191.
2. Keyes, R.W. and R. Landauer, 1970. Minimal energy dissipation in logic. IBM J. Res. Develop., pp: 152-157.
3. Bennett, C.H., 1973. Logical reversibility of computation. IBM J. Res. Develop., 17: 525-532.
4. Bennett, C.H., 1988. Notes on the history of reversible computation. IBM J. Res. Develop., 32: 16-23.

5. Perkowski, M., L. Jozwiak, P. Kerntopf, A. Mishchenko and A. Al-Rabadi *et al.*, 2001. A general decomposition for reversible logic. In: 5th Intl. Red-Muller Workshop, pp: 119-138.
6. Azad Khan, M.H., 2002. Design of full-adder with reversible gates. 5th ICCIT 2002, East West University, pp: 515-519.
7. Rafiqul Islam, M., M. Saiful, I.M. Rezaul and K.A. Al Mahmud, 2004. Synthesis of adder circuits using reversible Logic. In: Proc. 3rd IEEE Intl. Conf. for Upcoming Engineers, ICUE 2004, Ryerson University, Toronto, Canada.
8. Hasan, H.B., R. Islam, A.R. Chowdhury and S.M.A. Chowdhury, 2003. Reversible logic synthesis for minimization of full-adder circuit, Euro micro symposium on digital systems design, Belek-Antalya, Turkey, pp: 50-54.
9. Bruce, J.W., M.A. Thornton, L. Shivakumaraiah, P.S. Kokate and X. Li, 2000. Efficient adder circuits based on a conservative reversible logic date. IEEE Computer Society Annual Symposium on VLSI, Pittsburgh, Pennsytvania, pp: 83-88.
10. Rafiqul Islam,M.D., M.D. Saiful Islam, M. Rezaul Karim, A. Al Mahmud and H.M. Hasan Babu, 2004. Variable block carry skip logic using reversible gates. 10th Intl. Symp. on Integrated Circuits, Devices and Systems, ISIC-2004: Integrated Systems on Silicon Proceedings, Nanyang Technological University, Suntec, Singapore, pp: 9-12,.
11. Peres, A. 1985. Reversible logic and quantum computers. Physical Review A, 32: 3266-3276.
12. Kerntopf, P., 2000. A comparison of logical efficiency of reversible and conventional gates. 9th IEEE Workshop on Logic Synthesis, pp: 261-269.
13. Vos, D., 1997. Towards reversible digital computers. Proc. European Conference on Circuit Theory and Design, Budapest, pp: 923-931.
14. Toffoli, T., 1980. Reversible computing. Tech memo MIT/LCS/TM-151, MIT Lab for Comp. Sci.
15. Smolin, J. and D.P. Divincenzo, 1996. Five two-qubit gate are sufficient to implement the fredkin gate. Physical Review A, 53: 2855-2856.
16. Perkowski, M. *et al.*, 2003. A hierarchical approach to computer-aided design of quantum circuit. In: 6th Intl. Symp. on Rep. Methodol. Future Comp. Technol., pp: 201-209.