Generalized Construction of Quantum Multiplexers and De-Multiplexers Using a Proposed Novel Algorithm Based on Universal Fredkin Gate

Dipak Kumar Kole*,Jhuma Dutta†, Arpita Kundu‡, Suravi Chatterjee§, Suravi Agarwal#, Tanushri Kisku$
Department of Computer Science and Engineering
Jalpaiguri Government Engineering College
Jalpaiguri, India
*dipak.kole@gmail.com, †jhumadutta81@gmail.com, ‡kunduarpita2012@gmail.com, § choti4dgp@gmail.com,
#suravi.jgec@gmail.com, $tanushri.kisku@gmail.com

***Abstract*—In recent years, Reversible Logic is becoming a prominent technology due to its power reduction capability in circuit designing, thus having applications in quantum computing. The reversible circuit is implemented using reversible quantum gates such as CNOT gate, Fredkin gate, Toffoli gate etc. Among the various combinational circuits, quantum multiplexer and de-multiplexer circuits are two of the most important circuits. In this paper, we have proposed generalized algorithms for the construction of Quantum multiplexers (QMUX) and de-multiplexers (QDe-MUX) using universal Fredkin gate for any given number of select lines. Using these novel algorithms, any valid size of Quantum multiplexers (QMUX) and de-multiplexers (QDe-MUX) circuit can be generated.**

Keywords—Quantum computation; multiplexer; demultiplexer;

I. INTRODUCTION

Quantum mechanical effects have recently brought quantum logic circuits to a new dimensional attention. Its exponential speedups cover the fundamental limits of classical computation. As a result, quantum computation and information processing [1-2] remain an attractive area of research. One of the most desirable architecture requirements is to build an energy efficient circuit that has the potential of reducing power consumption[3], small and fast quantum computers. Unlike the classical logic gate operations, the operations on qubits must be reversible. A Reversible circuit/ gate generates unique output vector from each input vector, and vice versa, i.e., there is a one to one correspondence between the input and output vectors. The input values of a computation can be uncovered by the information on its outputs by Landauer's principle [4]. So during computation in reversible circuit there is no information loss. Most of the gates used in digital designs are not reversible, for example NAND, OR, and XOR gates. Only reversible gate in digital logic is the traditional NOT gate. Quantum circuit is a model for quantum computation in which a computation is a reversible transformation over a sequence of quantum gates. Each reversible gate has a cost associated with it called Quantum cost. The Quantum cost of a Reversible gate is the number of 2 x 2 reversible gates or quantum logic gates required in designing the required circuit. Garbage output is one of the important features of a reversible gate. It can be defined as the outputs of the gate which are not used as the inputs to another gate or as a primary output. These are used only to maintain reversibility of the circuit. Synthesis of reversible logic [5-7] is one of the major challenges in the quantum information processing domain and in the development of the architecture of the quantum computer. In quantum circuits, Toffoli gate [8], Fredkin gate [9] (both having 3 inputs and 3 outputs) are the popular universal as well as reversible quantum gates. Though some basic quantum logic circuits and gates have been demonstrated, still there are many scopes for designing efficient functional blocks such as quantum flip-flops, registers, multiplexers, de-multiplexers, adders etc. This paper deals with two of the combinational functional blocks, QMUX and QDe-MUX which can be constructed by physically realizable universal FREDKIN gate. Both multiplexer and de-multiplexer circuits have numerous applications in information processing and communication. In this paper we have proposed generalized algorithms for constructing both n: 1 QMUX and 1: n QDe-MUX depending on select lines, s where $n=2^s$ using universal Fredkin gates. The beauty of the proposed algorithms lies in the fact that we can design any generalized quantum multiplexer and de-multiplexer just by changing the parameter values. We have analyzed the algorithms in terms of time complexity. In Section II of this paper we have discussed some preliminaries about Fredkin gate, Multiplexer and De-multiplexer. Proposed Algorithms for generalized construction of QMUX and QDe-MUX have been stated in Section III. Illustration and analysis of the algorithms are discussed in Section IV. In section V results are analyzed in terms of performance metrics and comparing with existing methods. Finally conclusions are drawn in Section VI.

978-1-5090-2541-1/16/$31.00 ©2016 IEEE

II. PRELIMINARIES

The reversible logic gates should have n-inputs and n-outputs. In reversible logic, inputs can be uniquely recovered from the outputs. If a reversible gate has k inputs, and therefore k outputs, then it is a k x k reversible gate. Some important factors in reversible logic are Garbage Output, Constant Input, and Quantum Cost etc. The basic universal gates used for our proposed algorithms have been listed below.

A. Multiplexer

The word "multiplex" means many to one. The function of a MUX is to select one input among a group of inputs and pass the selected input to the output of the circuit. Basically, it consists of two types of inputs: one group is the data input and the other group is the select input and these select inputs decide which data input is to be passed to the output. A classical "*d: 1* MUX" implies a MUX circuit with d number of data input and one output. If there are s select lines, then the value of *d* is 2^s.A Multiplexer is also known as a data selector. The logic diagram of a 2:1 Multiplexer is shown in Fig. 1.

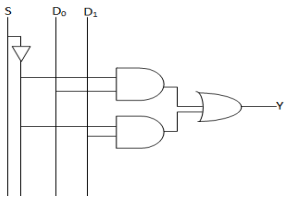

Fig. 1. Multiplexer

B. De-Multiplexer

The word "de-multiplex" means one to many. The function of a De-MUX is to take a single input and transmit it over several outputs. It consists of one input, a group of select lines and a group of outputs. A classical 1: d De-MUX implies a De-MUX circuit with 1data input and d number of outputs. If there are s select lines, then number of output lines is 2^s. The logic diagram of a 1:2 De-Multiplexer is shown in Fig. 2.

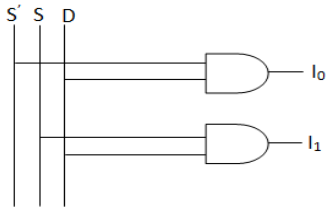

Fig. 2. De-Multiplexer

III. Proposed Method

In this section we have described the proposed generalized algorithms for logical construction and expansion of QMUX and QDe-MUX circuits using universal Fredkin gate. The algorithms make use of the fact that all circuits can be expanded logically if logic can be derived from simultaneous constructions of circuits. The algorithms are described in details in the following subsections.

A. QMUX Construction Algorithm

The algorithm stated below is used for constructing a quantum multiplexer circuit which has d data lines and a single output line depending on the number of select lines.

Input: Number of select lines (*s*), data lines (*d*)
Output: QMUX circuit with given number of data lines and select lines.

Step1: Read number of data lines (*d*), number of select lines(*s*)
Step2: If *d* is not equal to 2^s then QMUX design is not possible, Exit else continue the following steps
Step3: Set *m=d/2*; /* *m* is the number of required Fredkin gates for each stage */
Step4: Take *m* number of Fredkin gates. The inputs of the Fredkin gates are as follows:
First input: First select line
Second and Third input: Data line values in increasing order (D_0, D_1, D_2...)
Store the second output of each Fredkin gate
Step5: Update *s* as *s=s-1*
Step 6: Repeat while *s* is not equal to 0
i. Update *m=m/2*
ii. Take m Fredkin gates. The inputs of the Fredkin gates are as follows:
First input: Next select line.
Second and Third input: Any two of the stored outputs of the previous stage Fredkin gates (such that there is one input from each of the Fredkin gates and each of them is used uniquely),
Store the second outputs of the Fredkin gates. Remaining are garbage outputs.
iii. Update *s* as *s=s-1*
/* end of while */

B. QDe-MUX Construction Algorithm

The following algorithm is used for constructing a quantum de-multiplexer circuit which has a single data lines and a set of output lines depending on the number of select lines.

Input: Number of select lines (*s*), a single data line (*d*)
Output: QDe-MUX circuit with a set of outputs depending on select lines

Step1: Read a single data line (*d=1),* number of select lines (*s*)
Step2: Set *m=1* /* *m* is the number of required Fredkin gates for each stage*/
The inputs of the Fredkin gate are as follows:
First input: First select line
Second input: Data line
Third input: Constant 0
Store the second and third outputs of the Fredkin gate.
Step3: Update *s* as *s=s-1*
Step4: Repeat while *s* is not equal to 0
i. Update *m=m*2*
ii. Take *m* Fredkin gates. The inputs of the Fredkin gates are as follows:
First input: Next select line.
Second input: One of the stored outputs of the previous stage Fredkin gate(s) (such that each stored output is used only once)
Third input: Constant 0
Store the second and third outputs of the Fredkin gates. Remaining are garbage outputs.
iii. Update *s* as *s=s-1* /* end of while */
Step5: The second and third outputs of the final stage Fredkin

gates are the outputs of the QDe-MUX circuit.

Step6: End

IV. Analysis and Illustration

Analysis of the algorithms is done by calculating the time complexity. This section gives a detailed analysis of the algorithms we have stated in this paper and shows the illustration of the algorithms using the examples of 4:1 quantum multiplexer and 1:4 quantum de-multiplexer.

A. Time Complexity of QMUX Construction Algorithm

The time complexity of this algorithm is based on the number of select lines *s* taken as input from the user. Let us consider *T(s')* to be the time complexity for step number *s'*, where *s'* is a variable representing the number of select lines at a given step. *T(s')* represents the number of times the while loop is executed.

If $s' = 1$, $T(1) = c + T(0)$ (1)

where *c* is a constant.

If $s' = 2$, $T(2) = c + T(1)$

If $s' = 3$, $T(3) = c + T(2)$

.

.

If $s' = s$, $T(s) = c + T(s\text{-}1)$

Thus, the complexity is $O(s)$.

We know, $d = 2^s$ (2)

So, $s = \log_2 d$ (3)

Thus, we can write the time complexity of the QMUX construction algorithm as $O(\log_2 d)$.

B. Time Complexity of QDe-MUX Construction Algorithm

The time complexity of this algorithm is based on the number of select lines s taken as input from the user. Let us consider *T(s')* to be the time complexity for step number *s'*, where *s'* is a variable representing the number of select lines at a given step. *T(s')* represents the number of times the while loop is executed.

If $s' = 1$, then from (1), $T(1) = c + T(0)$

where *c* is a constant.

If $s' = 2$, $T(2) = c + T(1)$

If $s' = 3$, $T(3) = c + T(2)$

.

.

If $s' = s$, $T(s) = c + T(s\text{-}1)$

Thus, the complexity is $O(s)$.

From (2), we know, $d = 2^s$

So, $s = \log_2 d$

Thus, we can write the time complexity of the QDe-MUX construction algorithm as $O(\log_2 d)$.

C. Illustrations of the Algorithms

According to the algorithms stated above, the stepwise constructions of a 4:1Quantum multiplexer and 1:4 quantum de-multiplexer circuits are discussed below:

i. *Stepwise Construction of QMUX*

A multiplexer with n select lines and 2n data lines can be constructed step by step using the algorithm discussed above. There is a single output for all multiplexer circuits.

Construction of 4:1 Multiplexer

Step 1: Read d =4, s =2

Step2: d=2s, so MUX construction is possible, continue with the following steps:

Step 3: m =d/2 i.e., m =2

Step 4: Take m =2 Fredkin gates

Step 5: s =s-1, i.e. s =1, so Step 6 will be executed.

Step 6: Continue the while loop:

i. Update m =m/2 i. e.

ii. Take m Fredkin gates.

iv. s = s-1 i.e. s =0, so terminate the while loop

Step 7: The 2nd output of the last stage Fredkin gate is the output of the quantum multiplexer.

Step 8: End.

The inputs and outputs of the 4:1 Quantum Multiplexer is shown in Fig. 3.

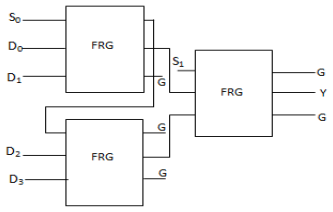

Fig. 3. 4:1 Quantum Multiplexer

In Fig.3, select lines are s_0 and s_1 and data lines are D_0, D_1, D_2, and D_3. The output of the circuit is,

$$Y = s_0' s_1' D_0 \oplus s_0 s_1' D_1 \oplus s_0' s_1 D_2 \oplus s_0 s_1 D_3 \quad (4)$$

Using the proposed algorithm we can construct any quantum multiplexer like 2:1, 4:1, 8:1 16:1 and so on. The circuit diagram for 8:1 Q MUX is shown in Fig. 4.

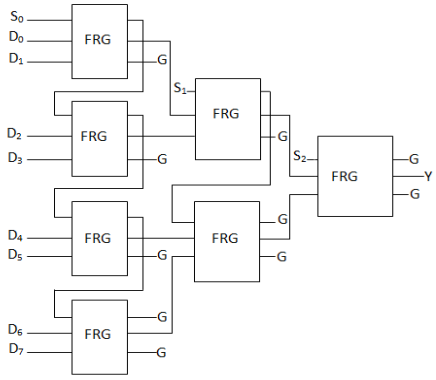

Fig. 4. 8:1 Quantum Multiplexer

ii. *Stepwise Construction of QDeMux*

A de-multiplexer with 1 data line and 2^n outputs can be constructed step by step using the algorithm discussed above.

Step 1: Read $d = 1$, $s = 2$

Step2: Take $m = 1$ number of Fredkin gates.

Step3: Update *s* as $s = s\text{-}1$, or $s = 1$ so while loop in step 4 will

be executed.

Step4: Update m =m*2 i.e. m =2.Take m =2 Fredkin gates. The second, third outputs of the previous stage Fredkin gate is the second input of the next stage Fredkin gates respectively. Update s as s =s-1, or s =0 so exit from while loop.

Step5: The second and third outputs of the final stage Fredkin gates are the outputs of the circuit.

Step6: End

The inputs and outputs of the 1:4 Quantum De-multiplexer is shown in Fig. 5.

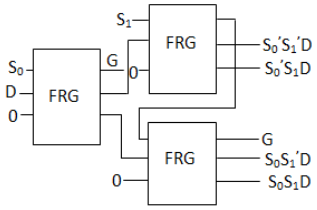

Fig. 5. 1:4 Quantum De-multiplexer

Using the proposed algorithm we can construct any quantum de-multiplexer like 1:2, 1:4, 1:8, 1:16 and so on. The circuit diagram for 1: 8 QDe-MUX is shown in Fig. 6.

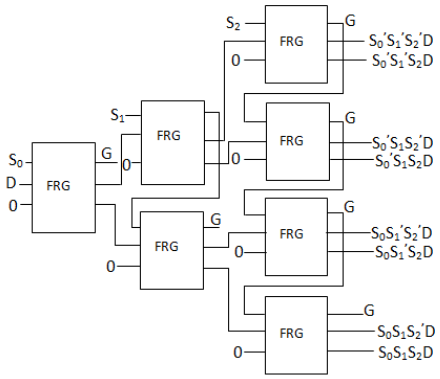

Fig. 6. 1:8 Quantum De-multiplexer

V. Results and Discussion

In this paper we have proposed algorithms for constructing generalized Q-Mux and QDe-Mux. The performances of the algorithms are analyzed in terms of performance metrics. The results are compared with existing methods and are shown by a comparison metrics in Table III.

A. Performance Metrics

The performance of any reversible quantum circuit is measured in terms of some basic parameters, like the number of gates present in the circuit, the total quantum cost of the circuit, the number of garbage outputs produced by the circuit, the total number of logical calculations performed by the circuit and so on. The generalized equations for the various parameters have also been discussed. Table I and Table II show the performance of the QMUX and QDe-MUX circuits designed using the generalized algorithms proposed in this paper. The hardware complexity [4] is measured by counting the number of AND operations, number of EX-OR operations and number of OR operations. Let

α = No. of EX-OR operations

β = No. of AND operations

δ = No. of NOT operations

Then the total hardware complexity T is given as sum of EX-OR, AND and NOT operations.

TABLE I. Performance Metrics of Q-mux

Design	Number of Gates	Number of Garbage Outputs	Quantum Cost	Total Logical Calculation
2:1 QMUX	1 FRG	2	5	$2\alpha+4\beta+2\delta$
4:1 QMUX	3 FRG	5	15	$6\alpha+12\beta+6\delta$
8:1 QMUX	7 FRG	10	35	$14\alpha+28\beta+14\delta$
D:1 QMUX	D-1 FRG	D+logD-1	5(D-1)	(D-1) $2\alpha+4\beta+2\delta$

TABLE II. Performance Metrics of QDe-mux

Design	Number of Gates	Number of Garbage Outputs	Quantum Cost	Total Logical Calculation
1:2 QDe-MUX	1 FRG	1	5	$2\alpha+4\beta+2\delta$
1:4 QDe-MUX	3 FRG	2	15	$6\alpha+12\beta+6\delta$
1:8 QDe-MUX	7 FRG	3	35	$14\alpha+28\beta+14\delta$
1:D QDe-MUX	D-1 FRG	log D	5(D-1)	(D-1) $2\alpha+4\beta+2\delta$

B. Comparison Metrics

The comparison metrics in Table III gives a pictorial view of the comparison of the proposed method with the existing ones in terms of design ratio, no. of gates used and quantum cost. The existing methods presented their works only for MUX whereas we have presented our work both for QMUX and QDe-MUX. From the results, we see that our proposed algorithms are comparable with other methods.

TABLE III. COMPARISON METRICS

Methods	Design	Number of Gates	Quantum Cost
Existing [10] Type1	4:1 MUX	6	21
Existing [10] Type2	4:1 MUX	4	19
Existing [11]	2:1 MUX	1	11
	4:1 MUX	9	109
	8:1 MUX	49	473
Existing [12]	2:1 MUX	1	5
	4:1 MUX	3	15
	8:1 MUX	7	35
Proposed Method	2:1 MUX	1	5
	4:1 MUX	3	15
	8:1 MUX	7	35
Proposed Method	1:2 QDe-MUX	1	5
	1:4 QDe-MUX	3	15
	1:8 QDe-MUX	7	35

VI. CONCLUSION

The main focus of the paper is towards the formulation of a generalized algorithm for constructing QMUX and QDe-MUX circuits using the FREDKIN gates. The proposed algorithms could be of great aid to design a QMUX and QDe-MUX circuit for any number of select lines only by changing the parameter values of the algorithms. QMUX and QDe-MUX circuits can further be used for designing other important Quantum circuits like Quantum ALU. We also have analyzed the quantum cost which is comparable with some existing work. In future, a more efficient algorithm can be derived, which can reduce the quantum cost of the circuits even more. Also, the same algorithm can be implemented using other quantum gates.

REFERENCES

[1] N. Gershenfeld and I. L. Chuang, "Quantum computing with molecules," Scientific American, June 1998.G.

[2] M. Nielsen and I. Chuang, "Quantum Computation and Quantum Information," Cambridge University Press, 2000.

[3] R. P. Feynman, Feynman Lectures on Computation. (A. J. G. Hey and R. W. Allen, Ed.), Perseus Books, USA, 1996.

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

[5] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes, "Synthesis of reversible logic circuits," IEEE TCAD, vol. 22, pp. 723-729, June 2003.

[6] D. Maslov, "Reversible Logic Synthesis," Ph.D Thesis, University of New Brunswick, Canada, September 2003.

[7] D. Maslov, G. Dueck, and N. Scott, Reversible Logic Synthesis Benchmarks: www.cs.uvic.ca/~dmaslov, 2004.

[8] T. Toffoli, "Reversible computing," Tech memo -MIT/LCS/TM-151, MIT Lab for Comp. Sci., 1980.

[9] E. Fredkin and T. Toffoli, "Conservative logic," Int. J. of Theoretical Physics, vol. 21, pp. 219-253, 1982.

[10] Y. Syamala and A.V.N Tilak, "Synthesis of Multiplexer and Demultiplexer Circuits using Reversible Logic," Int. J. of RecentTrends in Engineering and Technology, Vol. 4, No. 3, Nov 2010.

[11] Arijit Roy, Dibyendu Chatterjee and Subhasis Pal, "Synthesis of Quantum Multiplexer Circuits," IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 3, January 2012.

[12] Rakshith Saligram, Shrihari Shridhar Hegde, Shashidhar A Kulkarni, H R Bhagyalakshmi and M K Venkatesha, "Design of Fault Tolerant Reversible Multiplexer based Multi-Boolean Function Generator using Parity Preserving Gates," International Journal of Computer Applications (0975 – 8887), Volume 66– No.19, March 2013.