Chapter 2
Fundamentals of Reversible Logic

This chapter is devoted to the basic information and definitions required to travel around the reversible logic world safely. First, some preliminary concepts of reversible logic are introduced in Sect. 2.1. Basic and important definitions in reversible logic are explained in Sect. 2.2. Next the three most commonly used reversible gates of Feynman, Toffoli and Fredkin are presented in Sect. 2.3. The following Sect. 2.4 introduces interesting heuristics for reversible logic synthesis. Literature review of reversible logic synthesis methods is given in Sect. 2.5. As in everyday life everything has its price. The price of reversibility is the "garbage" outputs which are not allowed in quantum computing and have to be eliminated as explained in Sect. 2.6.

2.1 Preliminaries

Reversible computing is the path to future computing technologies, which all happen to use reversible logic. In addition, reversible computing will become mandatory because of the necessity to decrease power consumption.

Reversible logic circuits have the same number of inputs and outputs, and have one-to-one mapping between vectors of inputs and outputs; thus the vector of input states can be always reconstructed from the vector of output states [1]. Consequently, a computation is reversible, if it is always possible to uniquely recover the input, given the output. Each gate can be made reversible by adding some additional input and output wires if necessary [2].

Two constraints for reversible logic synthesis are: (1) feedback is not allowed, and (2) fan-out is not allowed (i.e., fan-out = 1). A gate with k inputs and k outputs is called a k*k gate. Several reversible gates have been proposed over the last decades.

S.M.R. Taha, *Reversible Logic Synthesis Methodologies with Application to Quantum Computing*, Studies in Systems, Decision and Control 37,
DOI 10.1007/978-3-319-23479-3_2

2.2 Basic Definitions

In this section some important factors in reversible logic are explained. The main object in reversible logic theory is the reversible function, which is defined as follows.

Definition 2.1 The function $f(x_1, x_2 \ldots x_n)$ of n Boolean variables is called reversible if:

1. the number of outputs is equal to the number of inputs.
2. any input pattern maps to a unique output pattern.

In other words, the output of a reversible function is a permutation of the set of its input [3, 4].

For an (n, k) function, i.e. function with n-input k-output, it is necessary to add inputs and/or outputs to make it reversible. This leads to the following definition.

Definition 2.2 "Garbage" is the number of outputs added to make an (n, k) function reversible. While the word "constant inputs" is used to denote the preset value inputs that were added to an (n, k) function to make it reversible. The constant inputs are known as ancilla inputs.

The relation between garbage outputs and constant inputs is [3, 4]

$$\text{input} + \text{constant input} = \text{output} + \text{garbage}$$

As with reversible gates, a reversible circuit has the same number of input and output wires; the reversible circuit with n inputs is called an n × n circuit, or a circuit on n wires. More generally, Fig. 2.1 illustrates the general reversible circuit of *temporary storage* [5, 6]. The top n–k lines transfer n–k signals Y to the corresponding wires on the other side of the circuit. The bottom k wires enter as the input value X and emerge as the output value f(X). These wires usually serve as an essential workspace for computing f(X). This circuit is said to compute f(X) using n–k lines of *temporary storage*. This leads to the following definition.

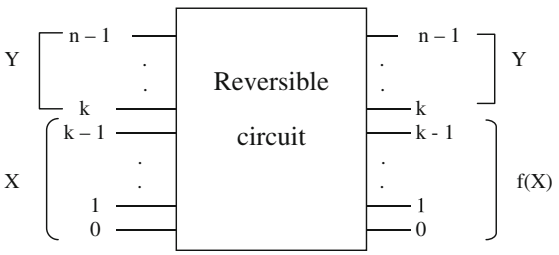

Fig. 2.1 Reversible circuit with n–k wires Y of temporary storage [5]

Definition 2.3 Temporary storage channels of reversible circuits are the redundant input–output line–pairs.

Other commonly used notations in reversible logic are explained as follows.

Definition 2.4 The size of a reversible gate is a natural number which shows the number of its inputs (outputs).

Definition 2.5 Balanced circuits are circuits for which each output value appears a number of times which is equal to the number of times that each of the other output values appears.

For example, in balanced binary logic the circuit has half of minterms with value 1. While in balanced ternary logic one third of minterms have value 0, one third have value 1 and one third have value 2.

Definition 2.6 Conservative circuits are circuits that have the same number of values in inputs and outputs, i.e. a conservative circuit preserves the number of logic values in all combinations.

In the next section it will be obvious that, while the (k, k) reversible gates: Wire (Buffer), Inverter, Swap, and Fredkin are balanced and conservative, other reversible gates: Feynman, and Toffoli are balanced but not conservative.

2.3 Reversible Logic Gates

Reversible circuits have functionality outputs and "garbage" outputs that are needed only to achieve reversibility. Figure 2.2 shows some of the binary (k, k) reversible gates that are commonly used in the synthesis of reversible logic circuits. While Wire (Buffer), Not, and Swap gates (Fig. 2.2a, b, and c, respectively) are naturally reversible, others are not, and thus "garbage" outputs have to be added.

Fredkin gate [7] together with Toffoli [8] and Feynman [9] gates belong to the most often discussed in reversible and quantum literature, the following sub-sections concentrate on the description of these gates.

2.3.1 Feynman Gate

The 2*2 Feynman gate (Fig. 2.2d), also called controlled-not (CNOT) or "quantum EXOR", realizes functions

$$P = A$$
$$Q = A \oplus B$$

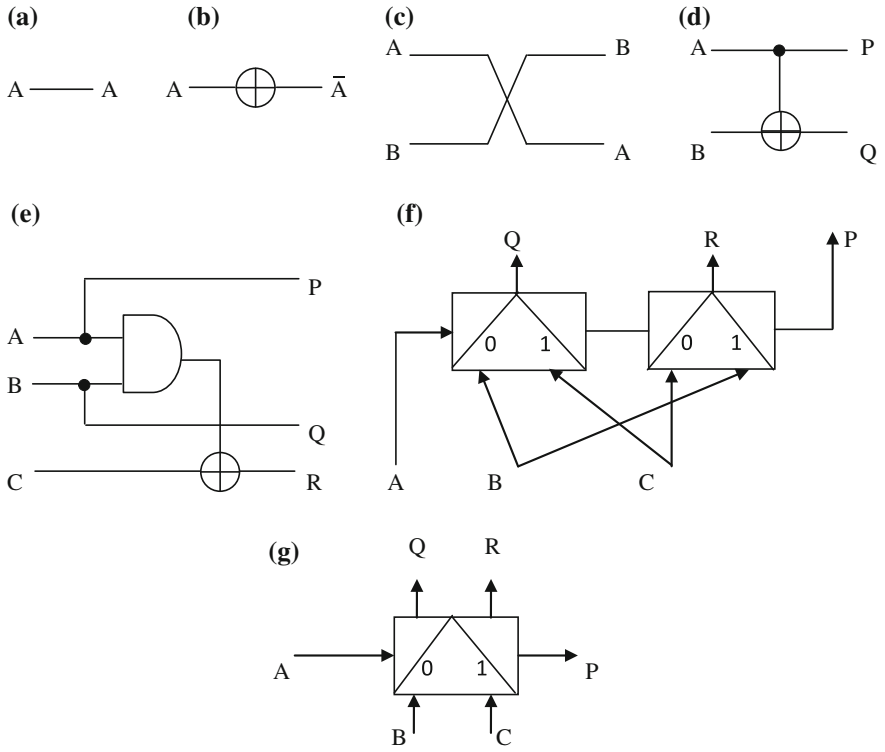

Fig. 2.2 Binary reversible gates: **a**. (1,1) Wire, **b**. (1,1) Inverter, c. (2,2) Swap, **d**. (2,2) Feynman gate, **e**. (3,3) Toffoli gate, **f**. (3,3) Fredkin gate, **g**. The simplified notation of the Fredkin gate

where A and B are the inputs, while P and Q are the outputs. The truth table of the Feynman gate is as follows:

A B	P Q
0 0	0 0
0 1	0 1
1 0	1 1
1 1	1 0

When A = 0 then Q = B, and when A = 1 then Q = $\bar{B}$ this is why it is called controlled not. With B = 0 Feynman gate is used as a fan-out gate or a copying gate (P = A and Q = A).

2.3.2 Toffoli Gate

The 3*3 Toffoli gate (Fig. 2.2e) is also known as 3*3 Feynman gate or controlled-controlled-not. It is described by the following equations:

$$\begin{aligned} P &= A, \\ Q &= B, \\ R &= AB \oplus C. \end{aligned}$$

The truth table of the Toffoli gate is as follows:

ABC	PQR
000	000
001	001
010	010
011	011
100	100
101	101
110	111
111	110

From the truth table, it can be seen that when A and B equal one then $R = \bar{C}$, this is why this gate is called controlled-controlled-not, because it has two control inputs A and B to invert the third input C. Toffoli gate is an example of *two—through* gates, because two of its inputs are given to the output.

2.3.3 Fredkin Gate

The 3*3 Fredkin gate (Fig. 2.2f) is also called controlled SWAP (CSWAP). It is described by the following equations:

$$\begin{cases} P = A \\ Q = C \quad \text{if } A = 1 \text{ else } B \\ R = B \quad \text{if } A = 1 \text{ else } C \end{cases} \quad \text{or,} \quad \begin{cases} P = A \\ Q = B \ \oplus AB \oplus AC \\ R = C \ \oplus AB \oplus AC \end{cases}$$

In terms of classical logic this gate is just two multiplexers in a flipped (permuted) way from the same control input A. Figure 2.2g shows the simplified notation of the Fredkin gate, while its truth table is shown below:

ABC	PQR
000	000
001	001
010	010
011	011
100	100
101	110
110	101
111	111

The 3*3 Fredkin gate is a permutation gate, it permutes the data inputs of its two multiplexers under control of the control input of these multiplexers. This control input is also an output from the Fredkin gate. Fredkin gates are called *one—through* gates, which means that gates in which one input variable is also an output.

2.4 Reversible Logic Synthesis

The real challenge in system design today, and in the future, is to design reliable systems that consume as little power as possible and in which the signals are processed and transmitted at very high signal integrity. Logically reversible devices have to be used in order to reduce (theoretically eliminate) power consumption [10]. Heuristics for smart method of reversible logic synthesis are the following [1, 11]:

- Do not create many outputs of gates and sub circuits.
- These outputs are reused as inputs to other gates.
- A good synthesis method minimizes the number of garbage signals.
- The total number of constants at inputs of the gates is kept as low as possible.
- If two copies of a signal are required, a copying circuit (Feynman gate) is used.
- The resulting circuit is acyclic which means that there can be no loops.
- The method must be generally applicable.

Since fan-out is not permitted, and assuming an appropriate technology, then a reversible logic circuit can realize the inverse specification simply by applying the gates in the reverse order. Hence, synthesis can be carried out from the inputs toward the outputs or from the outputs toward the inputs.

2.5 Overview of Reversible Logic Synthesis Methods

Here, the basic classifications with brief descriptions of reversible synthesis methods are done.

1. **Composition methods** [4, 12, 13]. The main idea is the use of small and well known reversible gates in composing a reversible block. A network is then synthesized by applying a conventional logic synthesis procedure.
2. **Decomposition methods** [4, 10, 14]. Decomposition methods can be considered as a top-down reduction of the function from its outputs to its inputs. A function is decomposed into several functions which are realized as separate reversible networks. There are several models of decomposition: Ashenhurst-Curtis (AC) decomposition, Modified Reconstructability Analysis (MRA), Bi-decomposition (BD), etc.
   The composition and decomposition methods can be multilevel, as well as they form a very general and powerful tool of logic synthesis [3].
3. **EXOR logic based methods** [4–6, 12, 13, 15–18]. These methods depend mainly on the use of Toffoli gates. The Toffoli gate uses the EXOR operation in its definition. The usage of the EXOR operation allows heuristic synthesis, as well as it is very hard to analyze. Thus only heuristic approaches currently work [3].
4. **Genetic algorithms** [10, 19, 20]. The general idea behind genetic algorithms is emulation of the evolution process. Evolutionary methods can be a good fit for the minimization of the general case of incompletely specified functions. This is important in order to realize smallest functional forms using the reversible structures. The genetic algorithm minimizer utilizes the Darwinian evolution, as well Lamarckian and Baldwinian evolutions to minimize the logic functions. These genetic algorithms have a lot of formulations and their actual implementations may vary. Their main weakness is their extremely bad scaleability [3].
5. **Search methods** [12]. The idea of circuit design is to take a circuit and start expanding and reducing it with maintaining its output functionality unchanged. After a certain number of such operations, a more compact circuit may be obtained. This technique is very expensive to use, since the size of the search space grows exponentially with an increase in the depth of the search.
6. **Group—theoretic methods** [21]. The group S_n of all permutations of n elements (objects) is a permutation group of order n!, where each of the n! group operators (permutations) is a group element. General k-cycle group representations for reversible circuits made of serial-interconnected and parallel-interconnected reversible primitives are done by performing the appropriate step-by-step permutations of each stage of the reversible circuit. One of the weaknesses of this approach is its need for a reversible specification.
7. **Synthesis of regular structures** such as **nets** [1, 10, 22]**, lattices** [10, 11, 23], and **RPGAs** [10, 22]. The idea behind these methods is to realize symmetric reversible functions in a regular structure of reversible gates. By a regular structure it is understood that a logic circuit and its physical layout structure being an array of identical cells regularly connected. Such methods have a high amount of garbage which is considered as a weakness of these methods.
8. **Spectral techniques** [24]. In these techniques a composition approach is applied and its spectral complexity is calculated. A spectral technique is used to find the best gate to be added in terms of gates (NOT, CNOT, Toffoli) and adds

the gate in a cascade-like manner. The output function is required to appear as a set of actual outputs, or their negations. Good results are obtained for small size reversible functions. A post processing process to simplify the network is used. The weaknesses of the method are: it scales badly, and requires a reversible specification [3].

9. **Exhaustive search** [5, 6]. Here, all optimal networks for 3 input reversible functions are found by matching all the minimal Toffoli gate networks (networks made of NOT, CNOT, Toffoli gates) with all reversible functions of 3 variables. It also considered minimal networks for 3 input reversible functions with Toffoli gate networks and SWAP gates. While the algorithm to synthesize optimal circuits scales better than its counterparts for irreversible computation, it is still limited by an exponentially growing search space.
10. **Binary Decision Diagram (BDD) based methods** [21, 23, 25, 26]. A Boolean function can be represented by BDD. A BDD is a directed acyclic graph where Shannon decomposition is often carried out in each node. In these methods BDDs are used to solve the quantified problem formulation. All minimal networks are found and the best one with respect to circuit complexity (cost) can be chosen, thus leading to cheaper realizations.

2.6 The Elimination of Garbage in Binary Reversible Circuits

In reversible logic circuits, where number of inputs is equal to the number of outputs, reversibility of computation has been achieved at the cost of introducing the constant inputs and garbage outputs (information that are not needed for the computation)

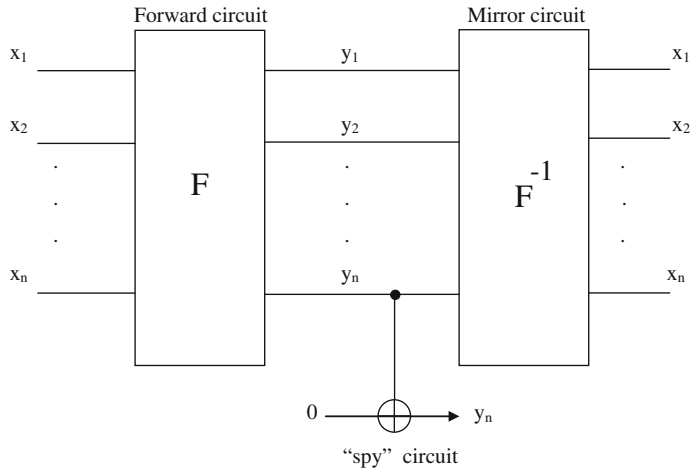

Fig. 2.3 The reversible circuit and its reversible mirror to eliminate garbage

[4, 12]. To eliminate the garbage outputs, for avoiding energy loss due to garbage accumulation, it is important to construct the inverse of the circuit. This is achieved by taking the outputs of the reversible circuit and producing from them "inversely" the inputs. This is important especially in quantum computing where garbage is not allowed. Figure 2.3 shows reversible circuit called "forward" (the block on the left), and its reversible inverse, called "mirror", (the block on the right).

Each reversible gate realizes a reversible function. That is, for each input pattern a unique output pattern, i.e. a one-to-one mapping, exists. Thus, calculating the inverse of the function F for an output pattern is essentially the same operation as propagating this pattern backwards through the circuit. Hence, if the cascade of n reversible gates $G = g_0\ g_1 \ldots g_{n-1}$ realizes a reversible function F, then the reverse cascade $G = g_{n-1}\ g_{n-2} \ldots g_0$ realizes the inverse function F^{-1}. The forward circuit is composed by using reversible gates, while the mirror circuit is composed by replacing each gate in the forward circuit by its inverse. It has been shown [10] that each of Fredkin, Toffoli, and Feynman gates is the inverse of itself. To measure the state of the hidden functions within the total network of the forward reversible part and the inverse reversible part, the "spy" circuit is used. The "spy" circuit is Feynman gate which is used as a copier by setting the value of the control input to value "0". The disadvantage of this approach of eliminating garbage signals is that it causes the duplication of the circuit's delay and the count of gates [11, 27].

References

1. M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola and B. Massey, Regular realization of symmetric functions using reversible logic, in *Proceedings of EUROMICRO Symposium on Digital Systems Design (Euro-Micro'01)*,Warsaw, Poland, pp. 245–252, September 2001
2. P. Kaye, R. Laflamme, M. Mosca, An Introduction to Quantum Computing. Oxford University Press Inc.,Oxford, 2007
3. D. Maslov, Reversible logic synthesis, Ph.D. thesis, The Faculty of Computer Science, The University of New Brunswick, Canada, 2003
4. D. Maslov, G.W. Dueck, Garbage in reversible designs of multiple output functions, in *Proceedings of the 6th International Symposium on Representations and Methodology of Future Computing Technologies (RM 2003)*, Trier, Germany, pp. 162–170, March 2003
5. V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, Reversible logic circuit synthesis, in *Proceedings of the International Conference on Computer Aided Design (ICCAD 2002)*, San Jose, California, USA, pp. 125–132, 10–14 November, 2002
6. V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, Synthesis of reversible logic circuits. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **22**(6), 710–722 (2003)
7. E. Fredkin, T. Toffoli, Conservative logic, Int. J. Theor. Phys. **21**(¾), 219–253 (1982)
8. T. Toffoli, Reversible computing, Technical Memo MIT/LCS/TM-151, MIT Lab. for Computer Science (1980). (Also, in Automata, Languages and Programming, (eds.) by W. de Bakker, J. van Leeuwen (Springer-Verlag, 1980), pp. 632–644.)
9. R.P. Feynman, Quantum mechanical computers. Opt. News **11**(2), 11–20 (1985)
10. A.N. Al-Rabadi, Reversible Logic synthesis: From Fundamentals to Quantum Computing. Springer, Berlin, 2004

11. A.T.S. Bashaga, Three-dimensional synthesis of ternary reversible lattice logic circuits, M.Sc. thesis, Department of Electrical Engineering, College of Engineering, University of Baghdad, 2007
12. G.W. Dueck, D. Maslov, Reversible function synthesis with minimum garbage outputs, in *Proceedings of the 6th International Symposium on Representations and Methodology of Future Computing Technologies (RM 2003)*, Trier, Germany, pp. 154–161, March 2003
13. D.M. Miller, D. Maslov, G.W. Dueck, A transformation based algorithm for reversible logic synthesis, in *Proceedings of the Design Automation Conference, DAC 2003*, Anaheim, California, USA, pp. 318–323, 2–6 June, 2003
14. M. Perkowski, L. Jozwiak, P. Kerntopf, A. Mishchenko, A. Al-Rabadi, A. Coppola, A. Buller, X. Song, M. Khan, S. Yanushkevich, V. Shmerko, A general decomposition for reversible logic, in *Proceedings of the 5th International Workshop on Applications of Reed-Muller Expansion in Circuit Design (Reed-Muller'01)*, Starkville, Mississippi, USA, pp. 119–138, 10–11 August, 2001
15. D. Maslov, G.W. Dueck, D.M. Miller, Synthesis of Fredkin-Toffoli reversible networks. IEEE Trans. Very Large Scale Integr. VLSI Syst. **13**(6), 765–769 (2005)
16. G.W. Dueck, D. Maslov, D.M. Miller, Transformation-based synthesis of networks of Toffoli/Fredkin gates, in *IEEE Canadian Conference on Electrical and Computer Engineering, CCECE 2003*, Montreal, Canada, May 2003
17. D. Maslov, G.W. Dueck, D.M. Miller, Fredkin/Toffoli templates for reversible logic synthesis, in *Proceedings of the International Conference on Computer-Aided Design (ICCAD 2003)*, San Jose, California, USA, 9–13 November, 2003
18. G. Yang, F. Xie, X. Song, W.N.N. Hung, M.A. Perkowski, A constructive algorithm for reversible logic synthesis, WCCI – 2006
19. M. Lukac, M. Pivtoraiko, A. Mishchenko, M. Perkowski, Automated synthesis of generalized reversible cascades using genetic algorithms, in *5th International Workshop on Boolean Problems*, Freiburg, Germany, pp. 33–45, 19–20 September, 2002
20. M. Lukac, M. Perkowski, H. Goi, M. Pivtoraiko, C.H. Yu, K. Chung, H. Jee, B. Kim, Y. Kim, Evolutionary approach to quantum and reversible circuits synthesis. Artif. Intell. Rev. **20**(3–4), 361–417 (2003)
21. A.N. Al-Rabadi, New classes of Kronecker-based reversible decision trees and their group-theoretic representation, in *Proceedings of the International Workshop on Spectral Methods and Multirate Signal Processing (SMMSP)*, Vienna, Austria, pp. 233–243, September 11–12, 2004
22. M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola, B. Massey, Regularity and symmetry as a base for efficient realization of reversible logic circuits, in *Proceedings of IWLS'01, Lake Tahoe*, California, USA, pp. 90–95, 12–15 June, 2001
23. A.N. Al-Rabadi, Spectral techniques in the reversible logic circuit synthesis of switching functions, in *Proceedings of the International Workshop on Spectral Methods and Multirate Signal Processing (SMMSP)*, Vienna, Austria, pp. 271–279, 11–12 September, 2004
24. D.M. Miller, Spectral and two-place decomposition techniques in reversible logic, in *Proceedings of the IEEE Midwest Symposium on Circuits and Systems (MWSCAS 02)*, II 493–II 496, August 2002
25. P. Kerntopf, A new heuristic algorithm for reversible logic synthesis, in *Proceedings of the 41st Annual Conference on Design Automation (DAC 2004)*, California, USA, pp. 834–837, 7–11 June, 2004
26. R. Wille, H.M. Le, G.W. Dueck, D. Groβe, Quantified synthesis of reversible logic, in *Design, Automation and Test in Europe (DATE 08)*, pp. 1015–1020, 2008
27. A.B. Khlopotine, M. Perkowski, P. Kerntopf, Reversible logic synthesis by iterative compositions, in *International Workshop on Logic Synthesis*, 2002

Springer

http://www.springer.com/978-3-319-23478-6

Reversible Logic Synthesis Methodologies with Application to Quantum Computing

Taha, S.M.R.

2016, XIII, 174 p., Hardcover
ISBN: 978-3-319-23478-6