UDC 004.3

REALIZATION OF IF-DECISION DIAGRAMS BY REVERSIBLE CIRCUITS

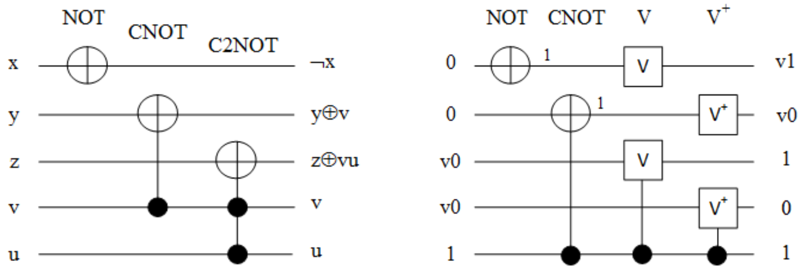
Prihozhy A.A.
Belarusian National Technical University, Minsk, Belarus
prihozhy@yahoo.com

Reversible circuits [1-6] are an abstraction of quantum circuits. All quantum circuits are reversible. The reversible circuit realizes a reversible function by means of reversible gates. The reversible function is bijective in nature since it has equal number of inputs and outputs, it maps each input to a unique output, and it is capable of reconstructing the input pattern from the output pattern. In the circuit, the same lines represent inputs and outputs, which are uniquely retrievable from each other. To maintain the reversibility, the circuit has no fan-out and feedback connections.

The reversible circuit is a cascade structure consisting of reversible gates from a gate library. The reversible gate has the form of $G(T, C)$, where $T \subset X$ is a target line, $C \subset X$ is a set of control lines ($C \cap \{T\} = \emptyset$), and X is a set of variables. The gate operation applies to the target line if the control lines meet true conditions. Syntactically the gate (target line) is represented by symbol $\bigoplus$, and its control lines are represented by symbol $\bullet$.

The reversible circuit describes a behavior by a superposition of three Boolean operations: $\neg$, $\wedge$ and $\oplus$. The Toffoli gate implements a vector Boolean function that maps three inputs to three outputs: $\mathrm{TG}(L, C_0, C_1) = (L\oplus(C_0 \wedge C_1), C_0, C_1)$ where $\oplus$ is Boolean exclusive or; $\wedge$ is Boolean conjunction; L is the Toffoli gate input at target line; C_0, C_1 are conditions at two control lines. The gate may have one control line, or may have no control lines at all. The Feynman gate with one control line realizes vector function $\mathrm{TG}(L, C_0) = (L\oplus C_0, C_0)$. Boolean inversion (not) of L is realized by a gate $\mathrm{TG}(L) =(\neg L)$ without control lines.

For the day, several reversible gate libraries are available. The NCT library [2] includes such fundamental reversible gates as NOT gate, CNOT (Feynman) gate with one control line, and C2NOT (Toffoli) gate with two control lines. The generalized GT library [6] includes additional multiple-control Toffoli gates. Figure 1a depicts the fundamental gates of the NCT library and the functions implemented by the gates.



a) b)

x	NOT(x)	V(x)	V+(x)
0	1	v0	v1
v0	v1	1	0
1	0	v1	v0
v1	v0	0	1

c)

Figure 1 – a) reversible gates of NCT library; b) quantum gates of NCV-|*v*1> library; c) operation of quantum gates

Quantum circuits carry out computations by manipulating quantum states of qubits. The qubit represents the state as $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. It assumes an arbitrary set of states. Quantum gates carry out an operation on qubit. Appropriate unitary matrices of $2^n\times2^n$ dimension describe the behavior of the gates.

Quantum logic gates are necessarily reversible in nature. A quantum circuit is a cascade of quantum gates. Several quantum gate libraries are available in the literature. The NCV library [4] is the most commonly used one for generating quantum circuits.

Work [6] introduced the extended NCV-|v_1> library. It allows for gates whose control lines may be sensitive to non-Boolean values (while e.g. the NCV gates considered above always require Boolean control lines).The library uses qudits instead of qubits and considers a 4-level (0, *v*0, 1 and *v*1) quantum system. Figure 1b depicts the quantum gates of the NCV-|v_1> library: NOT gate, controlled CNOT gate (both are similar to the corresponding reversible gates), not controlled and controlled V gate, and not controlled and controlled V^+ gate. Figure 1c describes the operation of the gates.

The quantum NOT gate and the controlled CNOT gate directly implement the corresponding reversible NOT gate without control and the CNOT gate with one control line respectively. At the same time, the quantum library does not contain a gate that directly implements the reversible C2NOT (Toffoli) gate (Figure 2a). For this reason, a cascade of quantum gates (as shown in Figure 2b) replaces the C2NOT gate when mapping the reversible circuit to a quantum circuit. This solution is expensive one; therefore, the research work has been done [6], which proposes an approach (Figure 2c) for reducing the overall cost of the quantum circuit.

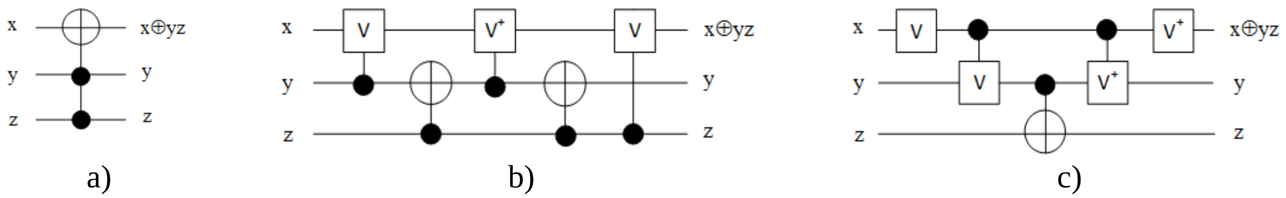


Figure 2 – a) reversible Toffoli gate; b) equivalent quantum circuit of NCV gates; c) equivalent quantum circuit of NCV-|v_1>gates

Binary decision diagrams (BDDs) are a well-known and widely used graph model (data structure) of Boolean functions [7, 8]. It lies in the basis of digital system modelling, synthesis and verification tools. Works [9, 10] propose a technique of synthesizing a reversible circuit from a function given as BDD. The technique substitutes all nodes of the BDD with a cascade of reversible gates. The size of reversible circuit directly depends on the BDD size; therefore, the circuit size can grow exponentially of the BDD inputs count.

The concept of if-decision diagrams (IFDs) as a graph representation of completely and incompletely specified Boolean functions was originally introduced in [11-14] as one of the results of the partial logic theory. IFD is an efficient model for the synthesis and optimization of digital systems. It is capable of representing and implementing parallel computations at logic level since its nonterminal nodes have three outgoing edges instead of two outgoing edges BDD's nodes have. Generally speaking, IFDs provide more universal and compact representation of Boolean functions against BDDs.

The capability of producing less cost reversible circuits for quantum implementation is a very important advantage of IFD. Using a four-root IFD of a three-bit adder as an example (Figure 3a), we introduce a technique of realizing the IFD by an appropriate reversible circuit (Figure 3b). The technique consists in the traversal of the IFD nodes from bottom to top, and automatically generating a cascade of reversible gates for each node, whose description includes the list of lines, their inputs and outputs, the target and control lines, and all the connections. For the IFD depicted in Figure 3a, the technique firstly passes nodes 1, 2 and 3 at bottom level, then it passes nodes 4, 5, 6 and 7 and at the middle level, and finally it passes nodes 9, 10, 11 and 8 at the top level.

The cascade configuration depends on the node type. Figure 4 shows cascades of reversible gates for six types of nodes the IFD of 3-bit adder is constructed of. Nodes 1, 2, 3, 9, 10 and 11 are of type *xor*(*d*, *g*) that is Boolean exclusive or, where *d* is a signal at the left daughter node and *g* is a signal at the middle (with complementation) and at the right daughter nodes. The corresponding cascade (Figure 4a) consists of only one reversible CNOT gate.

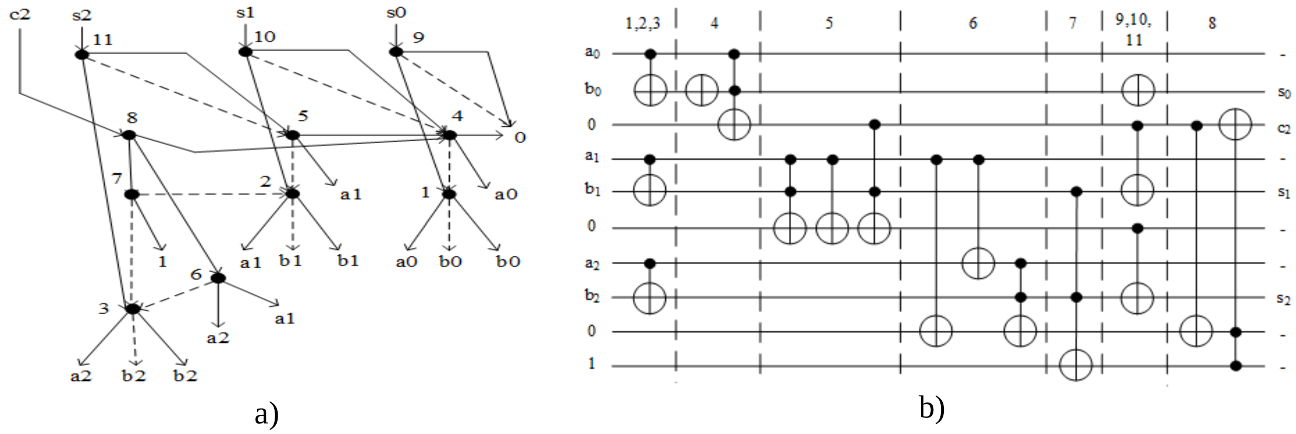


Figure 3 – a) four-root IFD of tree-bit adder (dash line is complementation); b) realization of the IFD by reversible circuit

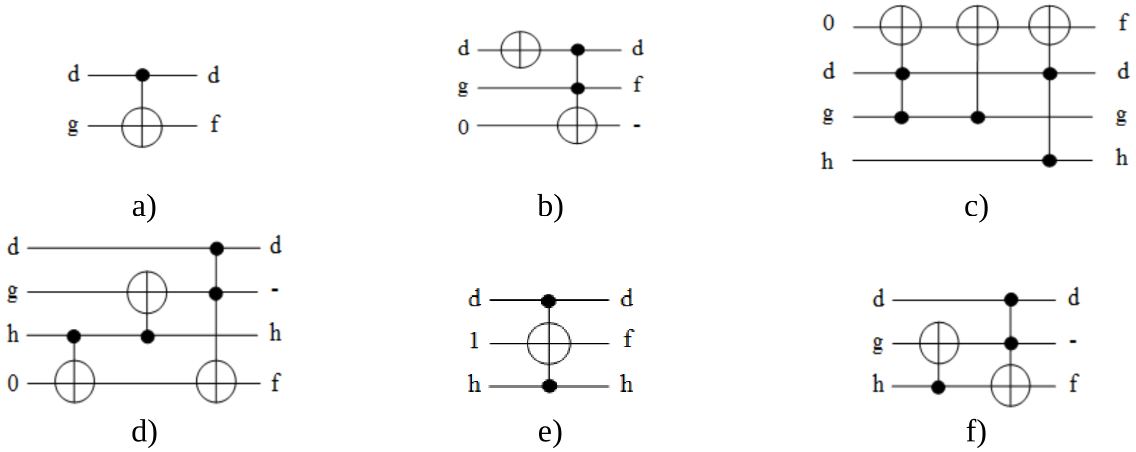


Figure 4 – Cascades of reversible gates for six types of IFD nodes: a) *xor*(*d*, *g*); b) *and*(¬*d*, *g*); c) *ifd*(¬*d*, *g*, *h*); d) *ifd*(¬*d*, *g*, *h*); e) *or*(¬*d*, ¬*h*); f) *ifd*(*d*, *g*, *h*)

Node 4 is of type *and*(¬*d*, *g*) that is Boolean conjunction with complementation of the first operand. The corresponding cascade consists of a NOT gate, a Toffoli gate and an ancillary line (Figure 4b).

Node 5 is of type *ifd*(¬*d*, *g*, *h*) that is Shannon expansion with complementation of the first operand, whose middle and right daughter nodes have the fan-out that exceeds 1. The corresponding cascade consists of a CNOT gate, two Toffoli gates and an ancillary line with input 0 as shown in Figure 4c.

Node 6 is of the same type *ifd*(¬*d*, *g*, *h*) as node 5 is, but its middle daughter node's fan-out equals 1. The corresponding cascade consists of two CNOT gates, a Toffoli gate and an ancillary line as shown in Figure 4d. It is simpler than the cascade shown in Figure 4c.

Node 7 is of type *or*(¬*d*, ¬*g*) that is Boolean disjunction with complementation of both operands. The corresponding cascade (Figure 4e) consists of a reversible Toffoli gate and an ancillary line.

Node 8 is of type *ifd*(*d*, *g*, *h*) that is Shannon expansion, whose middle and right daughter nodes' fan-out equals 1. The corresponding cascade (Figure 4f) consists of a CNOT gate and a Toffoli gate.

Figure 3b depicts the resulted reversible circuit of 10 lines and 17 gates. We have developed a reversible circuit simulation tool that have helped us to validate the circuit in Figure 3b. The technique and tool can realize other IFDs by reversible circuits. Work [13] proposes tools for analysis, transformation and optimization of the computations parallelization processes for many-processor architectures.

Conclusion. The binary decision diagram BDD is a powerful data structure that lies in the basis of numerous modern digital circuit modelling, simulation, synthesis and verification tools. The if-decision diagram IFD extends the capabilities of the binary decision diagram BDD due to its nonterminal nodes have three outgoing edges instead of two edges in BDD. Since the literature describes efficient techniques that map BDDs to reversible and further to quantum circuits, this paper studies how reversible circuits can realize IFDs while reducing the implementation cost. It proposes a technique of level-by-level traversal of a given IFD from bottom to top and replacing each nonterminal node by an appropriate cascade of reversible gates. The cascade cost significantly depends on the IFD node type and on the fan-out of all daughter nodes. The paper gives an example of realizing a 3-bit adder IFD by a reversible circuit constructed of the gates of NCV-$|v_1>$ library.

References

1. Bennett, C.H. Logical reversibility of computation / C.H. Bennett // IBM J. Res. Dev 17 (1973), pp. 525–532.

2. Toffoli, T. Reversible computing, in: W. de Bakker and J. van Leeuwen, editors / T. Toffoli // Automata, Languages and Programming, Springer, 1980, p. 632.

3. Fredkin, E.F. Conservative logic / E. F. Fredkin, T. Toffoli // International Journal of Theoretical Physics 21 (1982), pp. 219–253.

4. Barenco, A. Elementary gates for quantum computation / A. Barenco, C. H. Bennett, R. Cleve, D. DiVinchenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, H. Weinfurter // The American Physical Society, vol. 52, 1995, p. 3457-3467.

5. Nielsen, M. Quantum Computation and Quantum Information / M. Nielsen, I. Chuang // Cambridge Univ. Press, 2000.

6. Sasanian, Z. Realizing Reversible Circuits Using a New Class of Quantum Gates / Z. Sasanian, R. Wille, D. M. Miller // DAC 2012, June 3-7, San Francisco, 2012, p. 36-41.

7. C.Y. Lee, Representation of Switching Circuits by Binary-Decision Programs, Bell Systems Technical Journal, 1959, Vol. 38, No 4, pp. 985-999.

8. Bryant, R., Graph-based algorithms for Boolean function manipulation, IEEE Trans. on Comp. **35** (1986), pp. 677–691.

9. Wille R. Effect of BDD Optimization on Synthesis of Reversible and Quantum Logic / R. Wille, R. Drechsler // Electronic Notes in Theoretical Computer Science V. 253, 2010, pp. 57–70.

10. Zulehner, A. Accuracy and Compactness in Decision Diagrams for Quantum Computation / A. Zulehner, P. Niemann, R. Drechsler, R. Wille. // Design, Automation and Test in Europe Conference – DATE, 2019, pp.280-283.

11. Prihozhy, A.A. If-Diagrams: Theory and Application / A.A. Prihozhy // Proc. 7th Int. Workshop PATMOS'97. – UCL, Belgium, 1997. – P. 369 – 378.

12. Прихожий А.А. Частично определенные логические системы и алгоритмы / А.А. Прихожий / Минск, БНТУ. – 2013. – 343 с.

13. Прихожий А.А. Обобщение разложения Шеннона для частично определенных функций: теория и применение / А.А. Прихожий / Системный анализ и прикладная информатика. – 2013, № 1-2. – С. 6-11.

14. Prihozhy, A.A. Parallel Computing with If-Decision-Diagrams / A.A. Prihozhy, P.U. Brancevich // Proc. Int. Conference PARELEC'98. – Poland, Technical University of Bialystok. – 1998. – P. 179–184.

15. Prihozhy, A.A. Analysis, transformation and optimization for high performance parallel computing / A.A. Prihozhy // Minsk, BNTU. – 2019. – 229 p.