7-2003

Evolved Reversible Cascades Realized on the CAM-Brain Machine (CBM)

Andrzej Buller
ATR Human Information Science Labs

Marek Perkowski
Korea Advanced Institute of Science and Technology

Citation Details

Buller, Andrzej, and Marek Perkowski. (2003) "Evolved reversible cascades realized on the CAM-brain machine." Evolvable Hardware, 2003. Proceedings. NASA/DoD Conference on. IEEE, 2003. Advance online publication. DOI: 10.1109/EH.2003.1217675

Evolved Reversible Cascades Realized on the CAM-Brain Machine (CBM)

Andrzej Buller
ATR Human Information Science Labs
2-2-2 Hikaridai, Keihanna Science City
Kyoto 619-0288 Japan
tel. +81 774 95 1009, buller@atr.co.jp

Marek Perkowski
Department of Electrical Engineering and Computer Science, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Korea
mperkows@ee.kaist.ac.kr

Abstract

This paper presents a new approach to reversible cascade evolution based on a 3D cellular automaton. As a research platform we used the ATR's CAM-Brain Machine (CBM). Reversible circuits are investigated because they are expected to dissipate much less energy than their irreversible counterparts. One day they will be implemented as nano-scale 3-dimensional chips. A circuit is reversible if the number of its inputs equals the number of its outputs and there is a one-to-one mapping between spaces of input vectors and output vectors. This paper provides (1) a brief introduction to reversible logic concentrating on definitions and properties of the Feynman, Toffoli, Fredkin gates, (2) an introduction to the 3D Cellular Logic Machine (CLM) that is a cellular automaton with frozen and pulsing state variables, (3) a collection of reversible structures evolved using a dedicated GA and located in the CBM using the NeuroMaze 3.0 Pro, a software tool for computer-aided design of CBM-style structures.

1 Introduction

1.1 Importance of reversibility for future computers.

Reversible circuits appear to be a promising solution for most future nanotechnologies, because they are expected to dissipate much less energy than their irreversible counterparts exploited nowadays [1,18,20]. Reversibility is a necessary condition to generate no power at all during computing, and if the Moore Law continues to hold, the design of reversible circuits will become a necessity around year 2020 (conservatively estimating). The heat dissipation has two components, one related to circuit's technology that decreases every year and another one related to the information loss that is constant for irreversible technologies. When the first one will become smaller than the latter one, reversible design will become necessary to save power. It is speculated that three-dimensional Reversible Cellular Automata (3D CA) are the most efficient system architecture for future technologies [7] and it is believed that one day reversible architectures will be implemented as nano-scale 3-dimensional chips that nowadays are impossible because of, among others, the still unsolved problem of heat produced in traditional (irreversible) logic gates. Consult [7,20] for excellent modern overviews of reversible logic and related current and future circuit implementation technologies. Interestingly, as pointed out in [20], several existing computer architectures also require the design of reversible circuits [15,21]. A circuit is reversible if the number of its inputs equals the number of its outputs and there is a one-to-one mapping between spaces of input vectors and output vectors [18,22]. The qualification 'reversible' comes from the fact that every reversible gate or circuit provides unique deduction of the input vector based on the given gate definition and the output vector. During the development of Reversible Logic various basic sets of reversible gates were introduced. One of the sets, called the CNTS Library [20], includes the NOT gate that for 0 returns 1, while for 1 it returns 0, as well as the Feynman, Toffoli and SWAP gates that will be introduced in Section 2 together with the more complex Fredkin gate. In the present mainstream of the Reversible Logic-related research the circuits are being synthesized as cascades that can be drawn as arrays of separate horizontal lines representing wires with vertical symbols representing basic gates influencing locally the signals passing along the wires. Hence, fan-outs are not allowed. This restriction is followed in this paper. Cellular Automaton (CA) is defined as a computing device based on three elements: a set of connected sites (cells), a set of states that are allowed on the sites (cells), and a set of rules for how the states are updated ([8]: p. 102). For implementation of the reversible cascades we used CA adjusted based of the following assumptions [3]: (1) the state of every cell is defined using one binary variable called the pulsing state variable, as well as six binary variables called the frozen state variables, and (2) there is only one cell transition rule, that is the Boolean function S_1 that returns 1 when exactly one of its inputs is equal to one and returns zero otherwise. Thus, we are implementing arbitrary Boolean functions in reversible cascades which are next mapped to 3D CA. It has been speculated by several authors that future architectures, especially for brain building, will be evolved rather than designed and thus several evolutionary algorithms such as Genetic Algorithm and Genetic Programming have been already used to synthesize reversible and quantum logic circuits. In the frame of the presented research, a logic and layout of the desired cascade is generated by a specialized genetic algorithm (GA) developed in the Portland Quantum Logic Group [11-14] and then converted to certain states of the cells constituting the cellular automaton [3]. Our research is not related to "Morita's reversible CA" [16,17] because the employed CA itself is not reversible. This approach is being developed not to directly obtain reversible devices, but to obtain a platform for modeling large-scale reversible structures, mainly those used in brain building, but not only. The research will also serve the development of a future reversible CBM concept.

1.2 Toward reversible brain building.

The ATR's CAM-Brain Machine (CBM) is a dedicated FPGA-based hardware for experiments with 3-D CA designed to evolve and emulate large-scale para-neural networks [9]. Since the genetic algorithm located in the CBM proved to be too weak to be used for synthesis of useful structures, the ATR's Brain Building Group developed the NeuroMaze 3.0 Pro^{TM}, software for computer aided designing and testing of neural modules to be run on the CBM [10]. Thus the massive FPGA-based parallelism of CBM can be used not only for brain building but also for logic circuits emulation, especially for models presented in regular three-dimensional CAs. Because future generations of CBMs will have a massive computation power in small space, they should dissipate as little power as possible – using reversible logic can produce a dissipation-free brain (during calculation). Before the future Reversible 3D Cellular Automata - based CBMs will be built, one has to develop new methods for designing and simulating (emulating) their component subcircuits. This can be done in software, but the existence of NeuroMaze encourages us to use it also as a convenient tool for rapid modeling and testing of reversible cascades and 3D CA circuits. Hence, a number of reversible structures evolved using a dedicated GA developed in the Portland Quantum Logic Group [11-14] have been successfully converted into cellular-automatic structures and run on the CBM.

Summarizing, in order to have a desired reversible circuit run on a dedicated FPGA-based hardware (CBM), we (1) evolve the circuit using a special GA, (2) simplify the circuit using peephole optimizing transforms based on rules of EXOR algebra and tree search, (3) convert a code produced by the GA into cellular-automatic structure, (4) execute the structure in the CBM. Note that in our approach the evolution is only on the level of logic synthesis of complex logic gates and not on the low level cellular cells which approach would make the GA responsible for logic, timing, placement and routing. The approach proposed here combines evolutionary algorithm (EA) software, standard Computer Aided Design (CAD) and some human intervention, which seems to be a more realistic way to create complex circuits in the CBM than the entirely evolutionary approaches proposed earlier [5]. Reversible circuits with up to 20 gates have already been evolved [11-14]. Although the ATR CAM-Brain Machine [9] was used as the research platform, the methods of automated synthesis of reversible cascades in a cellular automaton presented here are general. This research is being conducted as a part of the Quantrix Project, launched as one of four themes explored in the framework of the Artificial Brain Project conducted at the ATR Human Information Science Laboratories, Kyoto [4]. It contains first results obtained in search for scientific grounds for a new evolvable hardware for on-board brains of intelligent robots. We want to reiterate that in no way we claim that our current approach saves power, on the contrary, reversible circuit emulated in FPGA is usually larger than an equivalent irreversible circuit and consumes more power. The sole goal of our approach is to emulate reversible 3D CA circuits that will be able to save power in future technologies used for brain building. Also, this paper is not related to simulating intelligent behaviors on CBM [2-5] since it is restricted only to the reversible CA aspect.

1.3 Nomenclature and lemmas

$\boldsymbol{x}'$ - Boolean negation ($0' = 1$, $1' = 0$)
$\boldsymbol{xy}$ – Boolean product ($00 = 0$, $01=0$, $10 = 0$, $11 = 1$)
$\boldsymbol{x} + \boldsymbol{y}$ – Boolean sum ($0 + 0 = 0$, $0+1 = 1+0 = 1$, $1 + 1 = 1$)
$\boldsymbol{x} \oplus \boldsymbol{y}$ - Exor (exclusive OR)
($0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$, $\boldsymbol{x} \oplus 1 = \boldsymbol{x}'$,
$\boldsymbol{x} \oplus 0 = \boldsymbol{x}$, $\boldsymbol{x} \oplus \boldsymbol{x}' = 1$)
$(\boldsymbol{x} \oplus \boldsymbol{y}) \oplus \boldsymbol{z} = \boldsymbol{x} \oplus (\boldsymbol{y} \oplus \boldsymbol{z})$, $\boldsymbol{x}(\boldsymbol{y} \oplus \boldsymbol{z}) = \boldsymbol{x}\boldsymbol{y} \oplus \boldsymbol{x}\boldsymbol{z}$,
$\boldsymbol{x} \oplus \boldsymbol{y} = \boldsymbol{y} \oplus \boldsymbol{x}$, $\boldsymbol{x} \oplus \boldsymbol{x} = 0$

2 Reversible Logic

This section introduces the basic reversible gates and provides examples of using them for synthesis of reversible cascades.

2.1 CNOT (Feynman Gate)

The CNOT (Controlled NOT) gate, called also Feynman gate, is represented using a compound of three symbols: ⊕, •, and | that represent an inverter, a control and a connection, respectively (Fig. 1a). It concerns two and only two wires. The logical value in the wire to which the control • is attached is the same both immediately before and immediately after the control. As for the wire on which the inverter ⊕ is attached, the CNOT's behavior depends on the value detected by the control. When the value detected by the control is 1, the gate affects the wire the same way as NOT. When the value detected by the control is 0, the logical value in the wire to which the inverter is attached is the same, both immediately before and immediately after the inverter (Fig. 1b). Since $\mathbf{0} \oplus \boldsymbol{y} = \boldsymbol{y}$ for any Boolean value of $\boldsymbol{y}$, for $\boldsymbol{x} = 0$ the Feynman gate behaves as a fan-out element.

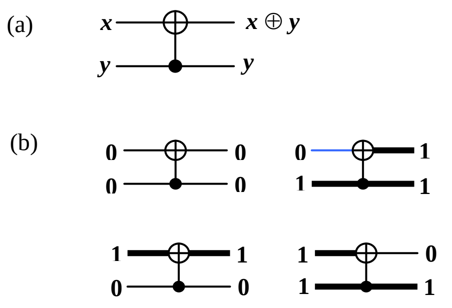

Figure 1. CNOT (Feynman) gate (a) symbol, (b) behavior.

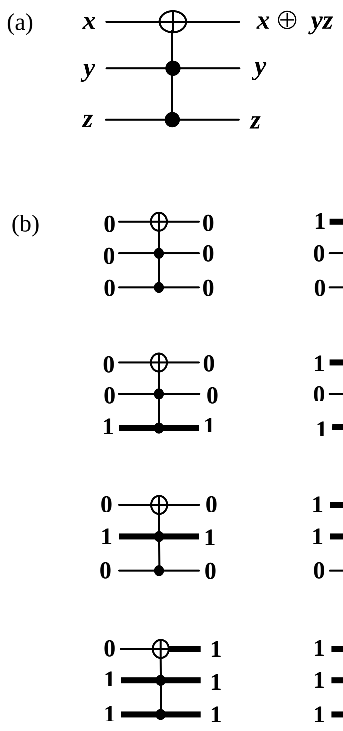

Figure 2. Toffoli gate (a) symbol, (b) behavior

2.2 Toffoli and Swap Gates

The Toffoli gate is represented as a composed of one inverter ⊕, two controls, and the vertical connection | (Fig. 2a). It concerns three and only three wires. The logical values in the wires to which the controls • are attached are the same both immediately before and immediately after a given control. As for the wire on which the inverter ⊕ is attached, the Toffoli's performance depends on the values detected by the controls. When the product of the values detected by the controls is 1, the gate affects the wire the same way as the NOT gate. When the product of values detected by the controls is 0, the logical value in the wire to which the inverter is attached is the same, both immediately before and immediately after the inverter. (Fig. 2b). The SWAP gate is represented as a compound of two copies of the symbol × and the vertical connection (Fig. .3). It swaps its input values.

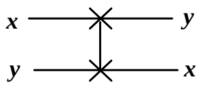

Figure 3. SWAP gate

2.3 CNTS Library and Reversible Cascades

The CNTS Library takes its name from the first letters of the names of gates: CNOT, NOT, Toffoli and SWAP [20]. It is an intellectual challenge to synthesize desired Boolean functions using exclusively the gates taken from the Library. A lot of effort in the field is devoted to the search of an efficient automation of Reversible Logic Synthesis (RLS). Genetic Algorithms and other heuristics are employed and some promising results are being reported [11-14]. The dominating trend in RLS is to arrange gates into cascades, i.e. arrays of horizontal wires interconnected using consecutive gates.

2.4 Constant inputs and garbage outputs

In the realm of Reversible Logic it is seldom possible to use as many inputs and outputs as in classic logic synthesis. There are three reasons. First, we may want to synthesize a function that by definition has a different numbers of inputs and outputs, usually real life functions have more inputs than outputs. While the basic requirement for reversible circuit is that a number of inputs is equal to the number of outputs. Second, even if the desired function has itself as many inputs as outputs, it may be not a reversible function and has thus to be converted to a reversible functions by adding the input signals (set to constant values) and the output signals (not used). The procedure for this conversion has been shown in [18]. The basic reversible gates used in such a new reversible circuit may produce some useful and some useless ouput values. These useless values are called garbage. It is one of the goals of reversible logic synthesis research to create systematic algorithms that will produce equivalent reversible circuits with as small garbage as possible. Sometimes the garbage of the entire circuit can be reduced via creating the so-called mirror circuit [18] but at the price of adding more intermediate wires. However, such an increase to the width of the circuit is often undesirable, for example when the reversible logic is implemented as a "quantum circuit" for which the width is nowadays restricted to seven in NMR technology. Therefore, a smart design is when the designer manages to make use of all outputs produced by the components of his circuit, thus introducing no input signals. The smaller the number of employed wires the better the design of a defined initial function in a reversible cascade. This task is quite difficult and different from standard logic synthesis. At present, no good methods exist for reversible synthesis of functions of many variables and high quality algorithms have been created only for few variable functions. Evolutionary algorithms are some of the most successful methods for reversible design so far, which is in contrast to the classical logic design, where evolutionary methods are not yet competitive to general purpose two- and many-level design tools that are capable of producing better-than-human designs for functions with hundreds of inputs and outputs and where they totally eliminate human logic minimization from modern industrial design processes. Thus, creating efficient reversible logic circuit synthesis approaches is a more practical challenge for evolvable hardware community than creating such approaches for standard irreversible circuits where these algorithms have little chance to compete with existing commercial CAD tools.

2.5 Fredkin Gate

The Fredkin gate is a 3-wire reversible device that can return various functions of selected input variables, including AND, OR, controlled SWAP and implication. Formally, it converts x, y and z into x, $xz \oplus x'y$, $xy \oplus x'z$, respectively (Fig. 4). Nevertheless, in order to note its useful properties, let us consider three cases when a given input is constant. The cases are shown in Figs. 5 and 6.

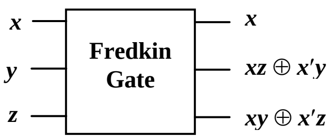

Figure 4. Fredkin Gate

Let x be set as a constant value. For x= **0**, the Fredkin gate will return **0**, $\mathbf{0}z \oplus \mathbf{1}y$ and $\mathbf{0}y \oplus \mathbf{1}z$, that equal to **0**, y and z, respectively. The calculations leading to this result are in Fig. 5. For x= **1**, the Fredkin gate will return **1**, $\mathbf{1}z \oplus \mathbf{0}y$ and $\mathbf{1}y \oplus \mathbf{0}z$, respectively, that equal to **1**, z and y, respectively. This way the Fredkin gate operates as a controlled SWAP gate (Fig. 5). Now let y has a constant value. For y= **0**, the Fredkin gate will return x, $xz \oplus x'\mathbf{0}$ and $x\mathbf{0} \oplus x'z$, that equal to x, xz and $x'z$, respectively. The calculations leading to this result are shown in Fig. 6. For y= **1**, the Fredkin gate will return x, $xz \oplus x'\mathbf{1}$ and $x\mathbf{1} \oplus x'z$, respectively. Observe that:

$$xz \oplus x'\mathbf{1} = xz \oplus x' = xz \oplus (x \oplus \mathbf{1}) =$$
$$= (xz \oplus x) \oplus \mathbf{1} = (xz \oplus x)' = (xz \oplus x\mathbf{1})' =$$
$$= (x(z \oplus \mathbf{1}))' = (xz')' = x' + z = x \Rightarrow z$$

while

$$x\mathbf{1} \oplus x'z = x \oplus x'z = (x' \oplus 1) \oplus x'z =$$
$$= (x'\mathbf{1} \oplus \mathbf{1}) \oplus x'z = (x'\mathbf{1} \oplus x'z) \oplus \mathbf{1} =$$
$$= (x'\mathbf{1} \oplus x'z)' = ((x'(\mathbf{1} \oplus z))' = (x'z')' = x + z.$$

This way the Fredkin gate appears to be a device that can return a Boolean product, a Boolean sum, as well as an implication. Although the Fredkin gate was invented as a primitive to be used in quantum computing, it can be built of primitives taken from the CNTS Library. Fig. 6 shows one of the solutions. Fedkin gates, as well as other shown here have been experimentally built in several future technologies, including nano, DNA and quantum. The solution from Fig..6, as well as many other useful reversible gate and circuit designs have been obtained using evolutionary programming system developed in the Portland Quantum Logic Group [11-14].

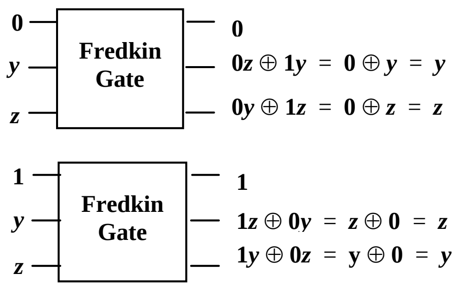

Figure 5. Fredkin Gate as a controlled SWAP

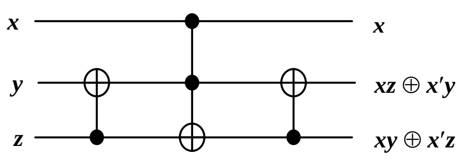

Figure 6. Fredkin Gate as a cascade of two Feynmans and one Toffoli.

3 CLM – a special cellular automaton for reversible modeling

As a medium for the modeled reversible computing we employed a special cellular automaton called Cellular Logic Machine (CLM) emulated on ATR's CAM-Brain Machine (CBM) [3]. CLM is 3-dimensional and works according to one simple rule. It can be imagined as a set of cubic cells arranged in such a way that each of the cubes has up to six neighbors. Every cell has a *door* in each of its 6 walls. The set of open doors and the set of closed doors must be determined in the framework of the automaton's initial state and kept unchanged for entire calculation process. Hence, the doors are called *frozen state variables*. Every cell can be either activated or not activated. Hence, the variable representing activation of a given cell is called the *activation* or *pulsing state variable*. A given cell gets activated in time t if and only if the number of its doors opened toward the neighbors activated in time t-1 is equal to 1. In order to describe the idea more precisely, let us employ the elementary symmetric function S_1 that returns 1 if and only if one of its six inputs is equal to one ([19], p. 99). Let us assume that binary sequence $a_1, a_2, \ldots, a_6$ represents activations of six neighbors of a given cell, while binary variables $d_1, d_2, \ldots, d_6$ are values at doors toward the neighbors. Let a_0 be activation of the cell itself. CLM has been adjusted to work in such a way that for every cell $a_{0,t+1} = S_1(d_1a_{1,t}, d_2a_{2,t}, \ldots, d_6a_{6,t})$.

3.1 Graphical representation of cell's state

A convenient way to show the state of a given cell uses a graphic planar representation. We propose the "arrow metaphor" where ▷, ◁, ▽, △, × and ○ represent open doors to Western, Eastern, Northern, Southern, Upper and Lower neighbor, respectively, all located in a square representing cell activation (pulsing state variable) (Fig. 7).

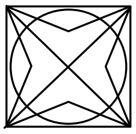

Figure 7. Planar representation of such state of CLM that all doors are open, while pulsing state variable (activation) is 0. ▷, ◁, ▽, △, × and ○ represent frozen state variables equal to 1, which can be interpreted as open doors to Western, Eastern, Northern, Southern, Upper and Lower neighbor, respectively [3]. Color of the square represents pulsing state variable (blank for activation 0, grey for activation 1).

3.2 Channel, Exor and Eeckhaut gate

A channel is an elementary structure of CLM. It employs only those cells that have only one gate open. Fig. 8 shows pulse propagation in a sample channel.

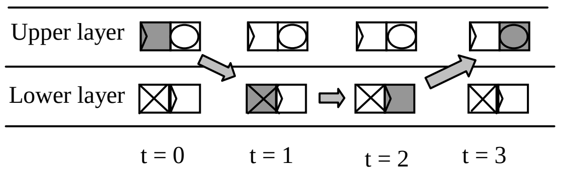

Figure 8. Pulse propagation in CA channel.

If a cell has two and only two gates opened, it can serve as an Exor gate (Fig. 9).

The Eeckhaut gate [6] serves as an AND gate (Figs. 10 and 11).

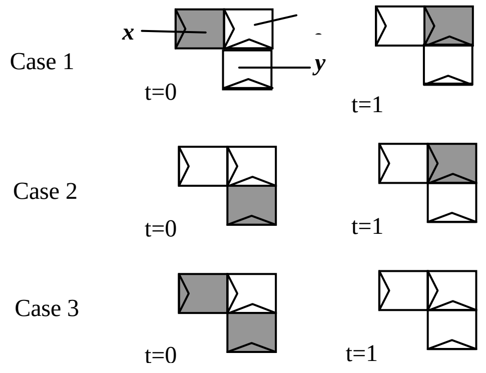

Figure 9. CLM-based Exor. $e_{t+1} = x \oplus y$. The trivial case $x = 0$, $y = 0$ is not shown since all cells would always be blank.

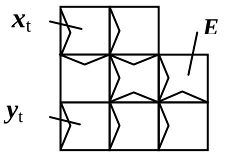

Figure 10. Eeckhaut gate works as delayed AND, i.e. $E_{t+3} = x_t y_t$.

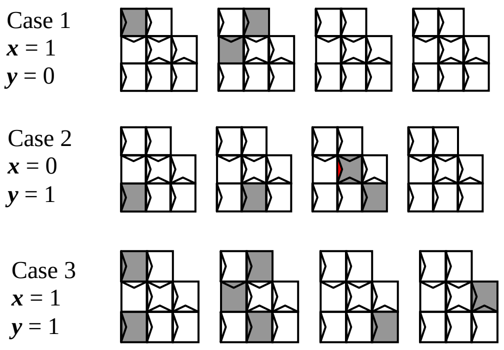

Figure 11. Behavior of the Eeckhaut gate

3.3 Reversible gates

Fig. 12a. shows a CA-based model of the Feynman gate (CNOT). Fig. 12b shows a model of the Tofoli gate using the Eeckhaut gate. Based on one Tofoli, two Feynman gates and two 4-cell channels, one can easily compose a cellular-automaton model of the Fredkin gate (Fig. 12c). The presented structures have been built under the NeuroMaze 3.0 Pro, a software tool for computer aided designing of 3-D β-PPNNs (Pulsed Para-Neural Networks) executable on the ATR's CAM-Brain Machine (CBM) [2]. Since the employed cellular automaton does not use all functions offered by the CBM, one of successor designs of the current CBM could be a reversible-logic-specific machine. It can be implemented in any technology in which the elementary reversible gates shown here can be built.

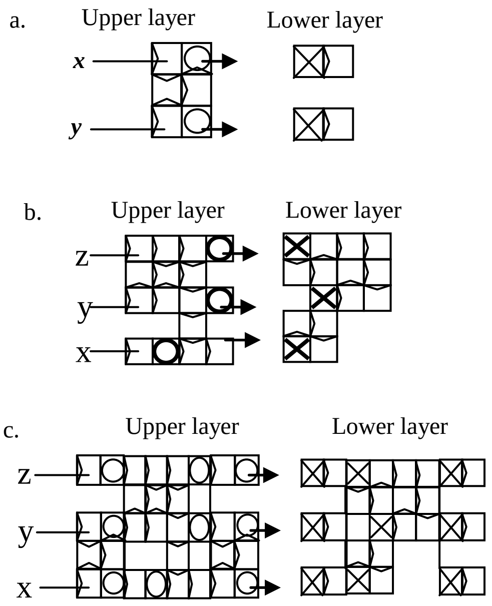

Figure 12. Reversible gates composed in CLM. (a) Feynman, (b) Toffoli, (c) Fredkin.

4 Concluding remarks

It was shown, that reversible cascades could be modeled in a special 3-dimensional cellular automaton with cells having a pulsing state variable, as well as a set of frozen state variables. For development and testing purposes, this automaton is executable on the ATR's CAM-Brain Machine (CBM). Since successful genetic algorithms for reversible logic synthesis have been built [11-14], the ATR's NeuroMaze 3.0 Pro, a software for computer aided designing of pulsed neural networks could be enhanced to facilitate fully automated creation of large-scale models of reversible cascades. Indeed, owing to regular input-output layouts of the presented reversible gates, they can be attached one to another by a simple program. Future work includes designing a universal reversible three-dimensional cellular architecture for brain building. With their cells implemented in (yet non-existent) nano-technologies, they will allow to create mobile robot's "computer brains" with extremely low power consumption.

Acknowledgements A. Buller's research is being conducted as a part of the *Research on Human Communication* supported by the Telecommunications Advancement Organization of Japan (TAO). M. Perkowski's research is supported by KAIST.

References

[1] C. Bennett, "Logical reversibility of computation, "*IBM J. Res. and Development, 17,* pp. 525-532, 1973.

[2] A. Buller, "CAM-Brain Machine and Pulsed Para-Neural Networks (PPNN): Toward a hardware for future robotic on-board brains," *Proc. 8th Int. Symp. on AROB,, Jan. 24-26, 2003, Beppu, Oita, Japan,* pp. 490-493.

[3] A. Buller, "Reversible Cascades and 3D Cellular Logic Machine," Technical Report TR-0012, ATR Human Information Science Laboratories, Kyoto, 2003.

[4] A. Buller, and K. Shimohara, "Artificial Mind. Theoretical Background and Research Directions," *Proc. 8th AROB,* pp. 506-509, 2003.

[5] H. de Garis, A. Buller, M. Korkin, F. Gers, N.E. Nawa, and M. Hough, "ATR's Artificial Brain ("CAM-Brain") Project : A Sample of What Individual "CoDi-1Bit" Model Evolved Neural Net Modules Can Do with Digital and Analog I/O," *Proc. 1st EHW,* July 19-21, Pasadena, California, pp. 102-110, 1999.

[6] H. Eeckhaut, and J. Van Campenhout, "Handcrafting Pulsed Neural Networks for the CAM-Brain Machine, " *Proc. 8th AROB,* pp. 494-501, 2003.

[7] M. Frank, http://www.cise.ufl.edu/~mpf and many pages linked there. This is the best resource for reversible logic, permanently updated.

[8] N. Gershenfeld, *The nature of mathematical modeling,* Cambridge: Cambridge Univ. Press, 1999.

[9] M. Korkin, G. Fehr, and G. Jeffrey, "Evolving hadware on a large scale," *Proc. 2nd NASA/DoD EHW, July 2000, Pasadena, USA,* pp. 173-181, 2000.

[10] J. Liu, *NeuroMaze User's Guide, Version 3.0,* ATR HIS, Kyoto, 2002.

[11] M. Lukac, and M. Perkowski, "Evolving Quantum Circuits Using Genetic Algorithms," *Proc. 4th NASA/DoD EHW, Wash. DC, USA,* pp. 173-181, 2002.

[12] M. Lukac, M. Pivtoraiko, A. Mishchenko, and M. Perkowski, "Automated Synthesis of Generalized Reversible Cascades using Genetic Algorithms," *Proc. 5th Int. Wk Boolean Pblms, Freiberg,* pp. 33-45, 2002.

[13] M. Lukac, M. Perkowski, and M. Pivtoraiko, "Evolutionary Approach to Quantum and Reversible Logic Synthesis," Subm to *Art Intel Rev J., 2003.*

[14] M. Perkowski, M. Lukac. M. Pivtoraiko, P. Kerntopf, M. Folgheraiter, H. Lee, W. Kim, W. Hwangbo, J-W. Kim, and Y-W Choi, "A Hierarchical Approach to Computer-Aided Design of Quantum Circuits," *Proc. 6th Intern. Symp. on Representations and Methodology of Future Computing Technologies,* Trier, Germany, March 10 -11, 2003.

[15] J.P. McGregor, and R.B. Lee, "Architectural Enhancements for Fast Subword Permutations with Repetitions in Cryptographic Applications," *Proc. ICCD,* pp. 453-461, 2001.

[16] K. Morita, and M. Harao, "Computation universality of one-dimensional reversible (injective) cellular automata," *Trans. IEICE, E-72,* pp. 758-762, 1989.

[17] K. Morita, and S. Ueno, "Computation-universal models of two-dimensional 16-state reversible cellular automata," *IEICE Trans. Inf. & Systems, E75-D,* pp. 141-147, 1992.

[18] M. Perkowski, A. Al-Rabadi, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, M. Azad Khan, A. Coppola, S. Yanushkevich, V. Shmerko, and L. Jozwiak, "A General Decomposition for Reversible Logic," *Proc. RM'2001, Starkville,* pp. 119-138, 2001.

[19] T. Sasao, *Switching Theory for Logic Synthesis,* Boston: Kluwer Academic Publishers, 1999.

[20] V.V. Shende, A.K. Prasad, I. Markov, and J.P. Hayes, "Reversible Logic Circuit Synthesis, "*Proc. 11th Intern. Workshop on Logic Synthesis,* pp. 125-130. 2002.

[21] Z. Shi, and R. Lee, "Bit Permutation Instructions for Accelerating Software Cryptography," *Proc. IEEE Intl. Conf. on App. Specific Systems, Architectures and Processors,* pp. 138 – 148, 2000.

[22] T. Toffoli, *Reversible Computing,* MIT/LCS/TM-151, MIT Lab for Comp. Science. 1980.