Cellular Automata realization of Regular Logic

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
http://www.atr.co.jp/his/~buller

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 cellular-automatic model of a reversible regular structure called Davio lattice. Regular circuits are investigated because of the requirement of future (nano-) technologies where long wires should be avoided. Reversibility is a valuable feature because it means much lower energy dissipation. 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. It is believed that one day regular reversible structures will be implemented as nano-scale 3-dimensional chips. This paper provides introduces the notion of the Toffoli gate and its cellular-automatic implementation, as well as an example of the Davio lattice built exclusively of Toffoli gates and run on a special cellular automaton called CAM-Brain Machine (CBM).

1. Introduction

This paper presents a cellular-automatic model of a reversible regular structure called Davio lattice. By a regular circuit we understand one that is composed of one or few types of identical logic/geometrical modules, connected only by abutting (short wires) and buses (long wires connected identically to all modules). We propose a new approach where a two-dimensional regular layout of a desired circuit is generated by software developed at Portland State University (Perkowski and Mishchenko 2002) and next converted to certain states of the cells constituting a cellular automaton. This requires a non-trivial three-dimensional layout of the lattice's modules such that signals produced in one module are received by another module in appropriate place and in appropriate time. The ATR CAM-Brain Machine was used as the research platform. The presented structure is the first step toward automated design of arbitrary combinational functions and finite state machines in cellular automata. Regular circuits seems to be the simplest way to realize logic in 3-dimentional layout. Reversible circuits appear to be a promising solution because they are expected to dissipate much less energy than their irreversible counterparts exploited nowadays (cf. Bennett 1973). Hence we combine these two ideas. It is believed that one day they will be implemented as nano-scale 3-dimensional chips that nowadays are impossible because of the still unsolved problem of heat produced in traditional logic gates.

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 (Fredkin & Toffoli 1982). 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. 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 (cf. Gershenfeld 1999: 102). For implementation of the reversible modules we used CA adjusted based of the following assumptions (Buller 2003): (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. This

approach is being developed not to directly obtain reversible devices, but to obtain a platform for modeling large-scale reversible structures.

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 (Korkin et al. 2000). 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™, software for computer aided designing and testing of neural modules to be run on the CBM (Liu 2002). This software appeared a convenient tool for rapid modeling and testing of reversible cascades.

2. EXOR Logic

Since the examples we provide are based on EXOR logic, the short summary of EXOR-related formulas may be helpful in reading this paper. They are

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

Exor Logic lemmas (Sasao 1999: 44)

$(x \oplus y) \oplus z = x \oplus (y \oplus z)$, $x (y \oplus z) = x y \oplus x z$, $x \oplus y = y \oplus x$, $x \oplus x = 0$

3. Reversible circuits

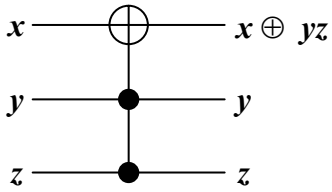
In the realm of Reversible Logic it is seldom possible to use as many inputs and outputs as in classic logic synthesis. There are tree reasons. First, we may want to synthesize a function that by definition has a different numbers of inputs and outputs, usually real life functions have more inputs than 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 input signals (set to constant values) and output signals (not used), as shown in (Perkowski et al 2001). The basic reversible gates used in such a new reversible circuit may produce some useful and some useless values. These useless values are called garbage. It is one of the goals of reversible logic synthesis technique to create systematic algorithms with as small garbage as possible. Sometimes the garbage of the entire circuit can be reduced via creating the so-called mirror circuit (Perkowski et al. 2001) but at the price of adding more intermediate variables. Nevertheless, increasing the width of the circuit is sometimes undesirable, for example when the reversible logic is to be implemented as a quantum computing device.

Thus, 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. So far no good methods exist for reversible synthesis of functions of many variables and high quality algorithms have been created only for few variables. 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.

3. Toffoli Gate

The Toffoli gate is a basic module of the Davio lattice we present here. It is represented on schemes as a compound of one inverter ⊕, two controls, and the vertical connection | (Figure 1a). 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 NOT. 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. (Figure 1b). The Toffoli gate is obviously reversible.

(a)



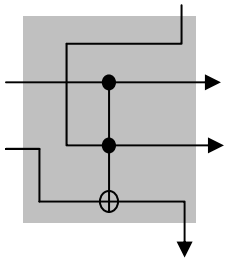
(b)

Case 1	*Case 2*	*Case 3*	*Case 4*
0 0 0 0 0 0	1 1 0 0 0 0	0 0 1 1 0 0	1 1 1 1 0 0
Case 5	*Case 6*	*Case 7*	*Case 8*
0 0 0 0 1 1	1 1 0 0 1 1	0 1 1 1 1 1	1 0 1 1 1 1

Figure 1. Toffoli gate (a) symbol, (b) illustration of behavior

4. Davio Lattice

Let us arrange inputs to and outputs from the Toffoli gate as shown below:



This way we obtained a functional "tile". Owing to the location of inputs and outputs, such tiles can be easily arranged into 2-dimensional regular structures. One of such structures is the Davio lattice (Figure 2).

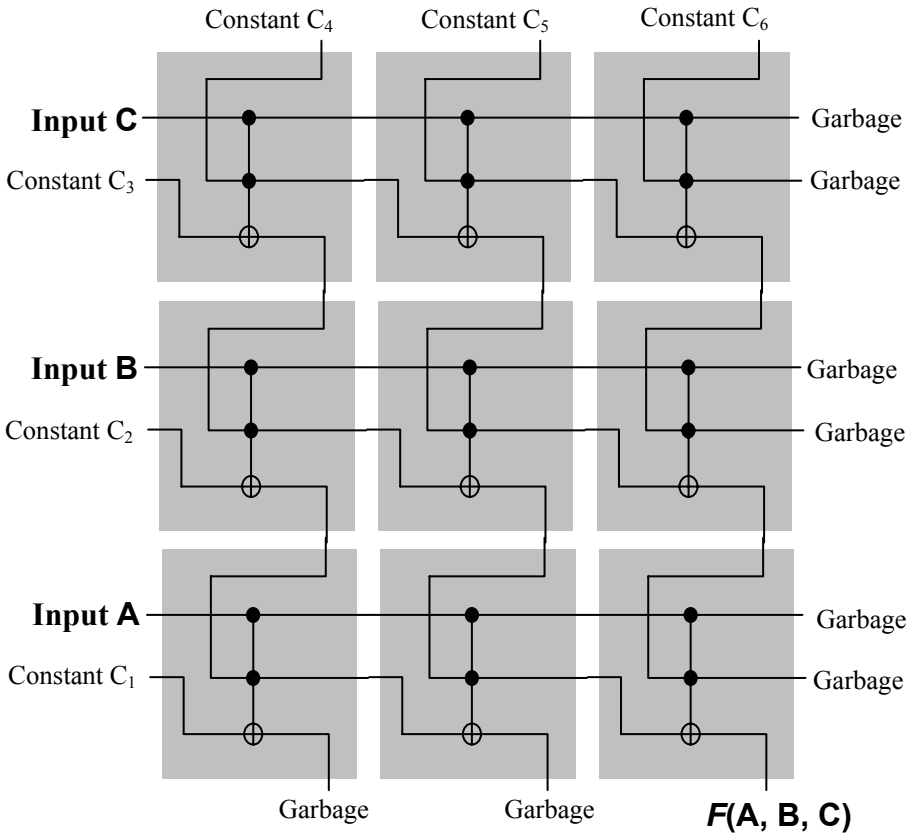


Figure 2. Davio lattice built exclusively of Toffoli gates

It can been proved that arbitrary Boolean function can be realized in a Reversible Davio Lattice, assuming repetition of variables in levels of the lattice. For instance, every symmetric function $F(A,B,C)$ of three variables A, B, C can be realized in the lattice from Figure 2, assuming correct setting of all constants C_i to values 0 and 1. If we take C5=1 and all other constants equal to zero, the lattice will return $F(A, B, C) = AB \oplus BC \oplus CA$.

5. Cellular Automaton (CA) for Reversible Modeling

As a medium for the modeled reversible computing Buller (2003) employed a cellular automaton (CA) run on ATR's CAM-Brain Machine (CBM). The cellular automaton 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 *activation* or *pulsing state variable*. A given cell gets activated in time t, if and only if the number of its doors opened toward neighbors activated in time t-1 is equal to just 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 (Sasao 1999: 99). Let us assume that binary a_1, a_2, …, a_6 represent activations of six neighbors of a given cell, while binary variables d_1, d_2, …, d_6 are doors toward the neighbors. Let a_0 be activation of the cell itself. The cellular automaton has been adjusted to work in such a way that for every cell $a_{0,t+1} = S_1(d_1a_{1,t}, d_2a_{2,t}, \ldots, d_6a_{6,t})$.

5.1. Graphic representation of cell's state

A convenient way to show a state of a given cell using 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) (Figure 4.1). Let color of the square be white or red for activation 0 or 1, respectively.

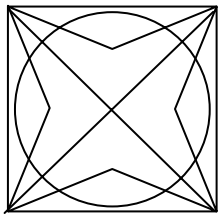


Figure 3. Graphic planar representation of such state of CA that all doors are open, while pulsing state variable (activation) is 0. ▷, ◁, ▽, △, × and ○ represent frozen state variables equal to 1, which can be interpreted as open doors to Western, Eastern, Northern, Southern, Upper and Lower neighbor, respectively. This figure was proposed by Michal Joachimczak as a notation in computer aided design of cellular-automatic structures. Color of the square represents pulsing state variable (white for activation 0, red for activation 1).

5.2. Channel and Exor

Channel is elementary structure of the discussed CA. It employs only cells that have only one gate open. Figure 4 and 5 show two samples of propagation of activation in the channels. If a cell has two and only two gates opened, it can serve as Exor gate (Fig. 6).

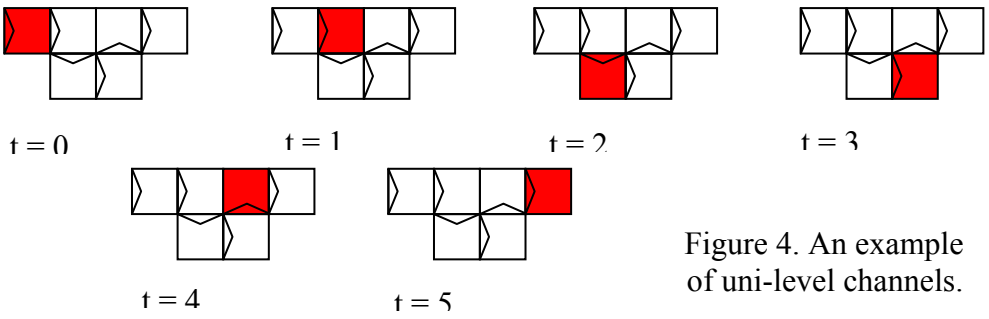


Figure 4. An example of uni-level channels.

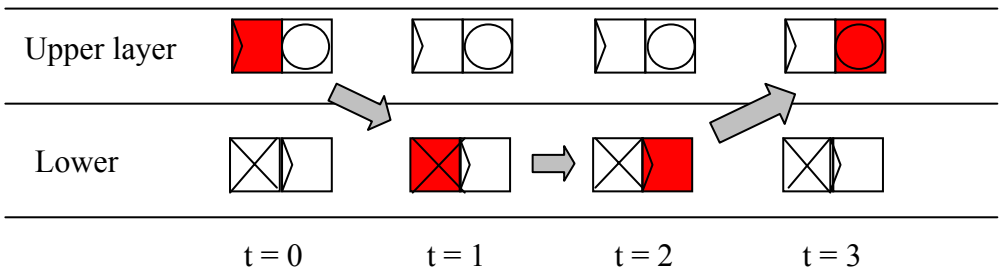


Figure 5. An example of multi-level channels.

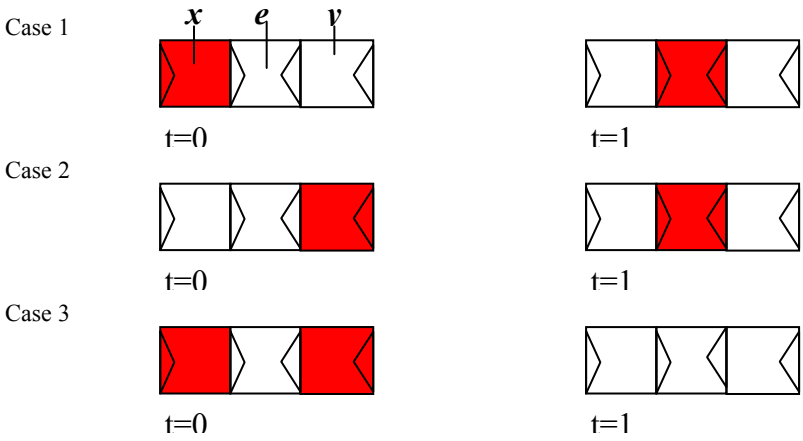


Figure 6. CA-based Exor. $e_{t+1} = x \oplus y$. The trivial case $x = 0, y = 0$ is not shown since all cells would always be blank.

5.4. Eeckhaut gate

Eeckhaut gate (Eeckhaut & Van Campenhout 2003) serves as an AND gate (Fig. 7 and 8).

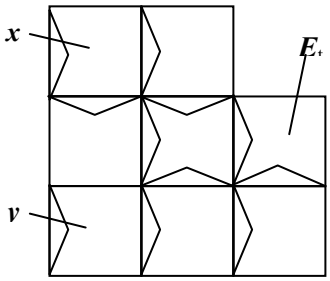


Figure 7. Eeckhaut gate which works as delayed AND, i.e. $E_{t+3} = x_t y_t$.

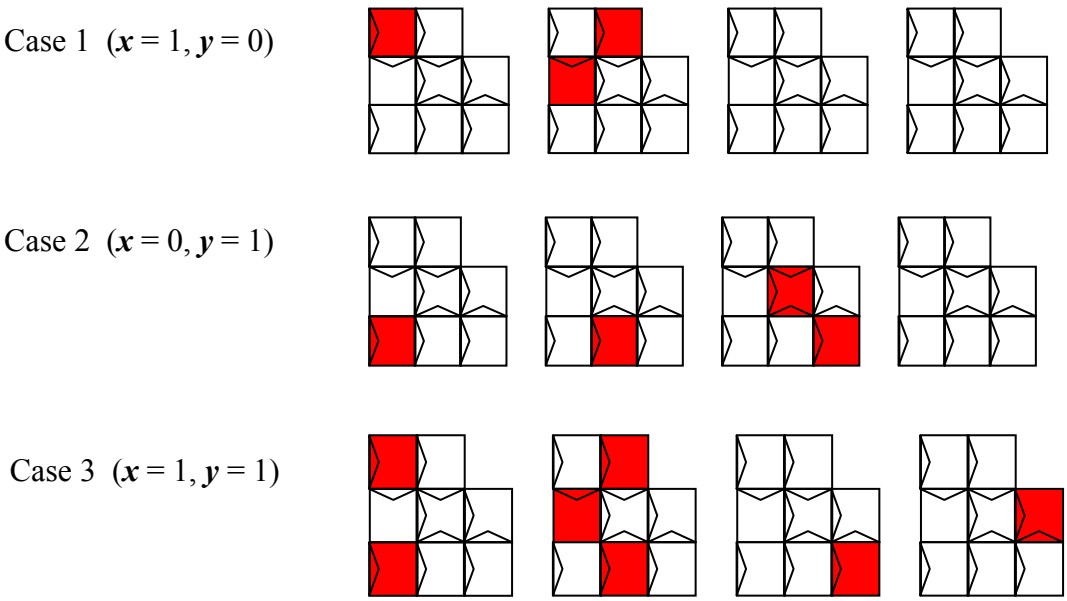


Figure 8. Behavior of the Eeckhaut gate

6. Cellular Toffoli gate

Figure 9 shows a cellular model of Tofoli gate using an Eeckhaut gate. Figure 10 shows the model of Tofoli gate modified to serve as a tile in the Davio lattice. Note that the layout of cells is such that for each "wire" the propagation time is the same and is equal to 18 clocks. The structure 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) (Buller 2003a).

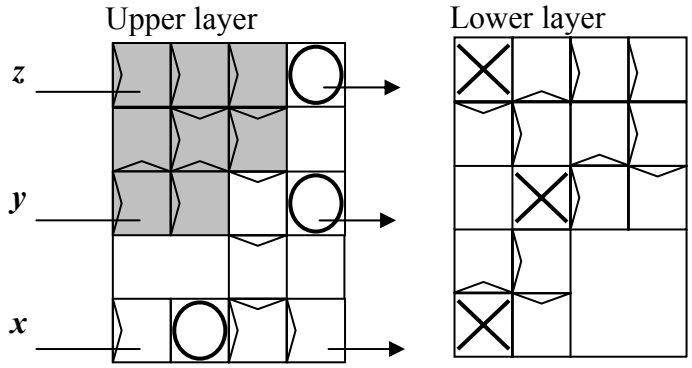


Figure 9. Toffoli gate modeled in CA. One of components is Eeckhaut gate (grey color)

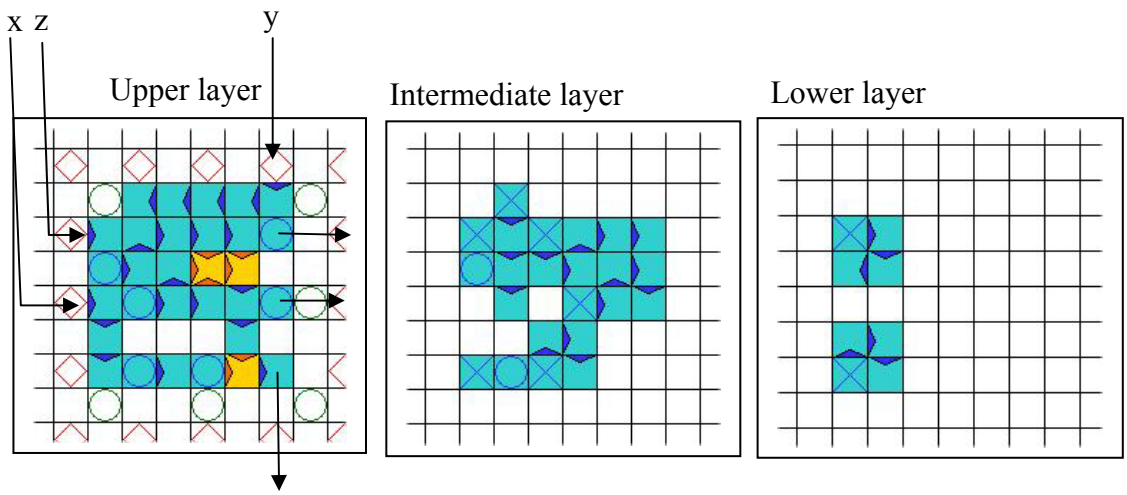


Figure 10. Cellular Toffoli gate modified to serve as a tile in the Davio lattice. The propagation time of 18 clocks is equal for each "wire".

7. Caellular Davio lattice

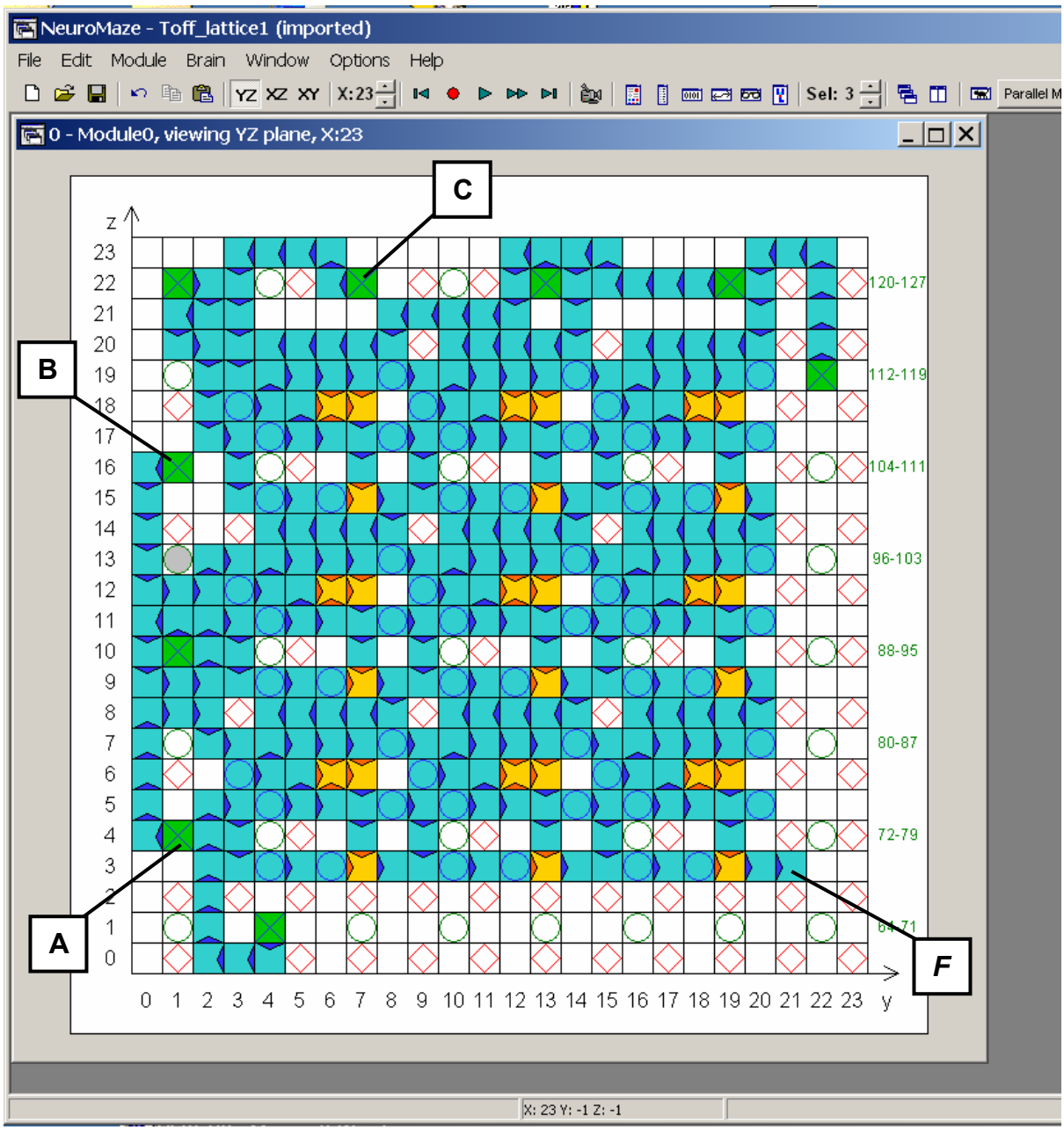


Figure 11. Davio lattice's upper layer in the NeuroMaze's worksheet.

8. Conclusions and future work

It was shown, that the Davio lattice could be modeled on in a 3-dimensional cellular automaton where some cells's state variables were set as "frozen". This automaton is executable on the ATR's CAM-Brain Machine (CBM). 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 structures, they can be attached one to another by a simple program. Several other regular logic structures presented by our group in the past can be mapped to 3-dimensional cellular automata in a quite similar way. The future research will include extending the algorithm for synthesis of lattices to multiple-output circuits and automating the 2D-3D transformation to avoid buses.

Acknowledgement The research is being conducted as a part of the *Research on Human Communication* supported by the Tele-communications Advancement Organization of Japan (TAO). Marek Perkowski has research support from KAIST.

Notes

An example of regular circuits are lattices, proposed for the first time in {Perkowski & Pierzchala 1993, Pierzchala et al 1994} Lattices can be generated for binary, multiple-valued and continuous (including fuzzy and Lukasiewicz) logics (Pierzchala and Perkowski 1999) as well as for various technologies, such as single electron transistor (Hasegawa 2001, Hasegawa et al 2001, Postma 2001) or quantum (Al-Rabadi 2002) For each of these logics and realization technologies a cellular automaton as presented below can be created, but here for simplicity we illustrate only the binary case. Much research on 2-dimensional regular combinational circuits has been published previously (Perkowski et al 1997, Chrzanowska-Jeske et al 1999) and efficient software to generate various types of lattices has been written (Perkowski and Mishchenko 2002). Three-dimensional lattices have been proposed also, both for irreversible binary and ternary logic (Perkowski et al 1997), and for reversible logic (Perkowski et al 2001, Al-Rabadi 2002). However, all these designs assumed the presence of buses – i.e. long wires in which the signal is propagated with very small delay. These buses are used for all input variables. Therefore these designs cannot be directly adapted for cellular automata model where buses do not exist and all communication is therefore from cell-to-cell only.

References

1. Al-Rabadi, A (2002) Novel Methods for Reversible Logic Synthesis and Their Quantum Computing, *Ph.D. Thesis*, Portland State University, November 2002.
2. Bennett C (1973) Logical reversibility of computation, *IBM Journal of Research and Development, 17*, 525-532.
3. Buller A (2003a) CAM-Brain Machine and Pulsed Para-Neural Networks (PPNN): Toward a hardware for future robotic on-board brains, *Proceedings of the Eighth International Symposium on Artificial Life and Robotics (AROB 8th '03), January 24-26, 2003, Beppu, Oita, Japan*, 490-493.
4. Buller A (2003b) Reversible Cascades and 3D Cellular Logic Machine, Technical Report TR-0012, ATR Human Information Science Laboratories, Kyoto.

5. Buller A, Shimohara K (2003) Artificial Mind. Theoretical Background and Research Directions, *Proceedings of the Eighth International Symposium on Artificial Life and Robotics (AROB 8th '03), January 24-26, 2003, Beppu, Oita, Japan*, 506-509.
6. Chrzanowska-Jeske M, Xu, Perkowski, M (1999) Logic Synthesis for a Regular Layout. *VLSI Design: An International Journal of Custom-Chip Design, Simulation, and Testing*.
7. de Garis H, Buller A, Korkin M, Gers F, Nawa NE & Hough M (1999) 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, *Proceedings of The First NASA / DoD Workshop on Evolvable Hardware*, July 19-21, Pasadena, California, 102-110.
8. Eeckhaut H, Van Campenhout J (2003) Handcrafting Pulsed Neural Networks for the CAM-Brain Machine, *Proceedings of the Eighth International Symposium on Artificial Life and Robotics (AROB 8th '02), January 24-26, 2002, Beppu, Oita, Japan*, 494-501.
9. Fredkin E, Toffoli T (1982) "Conservative Logic," *Int. J. of Theoretical Physics,* 21, pp. 219 – 253.
10. Gershenfeld N (1999) *The nature of Mathematical Modeling*, Cambridge: Cambridge Univ.Press.
11. Hasegawa H (2001) Quantum Devices and Integrated Circuits Based on Quantum Confinement in III-IV Nanowire Networks Controlled by Nano-Schottky Gates," *Proc. ECS Joint Int. Meeting and Sixth Int. Symp. on Quantum Confinement*, San Francisco.
12. Hasegawa H, Ito A, Jiang C, Muranaka T (2001) Atomic Assisted Selective MBE Growth of InGaAs Linear and Hexagonal Nanowire Networks for Novel Quantum Circuits, *Proc. of the 4th Int. Workshop on Novel Index Surfaces (NIS'01),* Apset, France..
13. Postma H.W, Teepen T, Yao Z, Grifoni M, Dekker C (2001) Carbon Nanotube Single-Electron Transistors at Room Temperature, *Science*, Vol. 293, No. 5527, 76-79..
14. Korkin M, Fehr G, Jeffrey G (2000) Evolving hadware on a large scale, *Proceedings, The 2nd NASA/DoD Workshop on Evolvable Hardware, July 2000, Pasadena, USA*, 173-181.
15. Liu J (2002) *NeuroMaze User's Guide, Version 3.0*, ATR HIS, Kyoto.
16. Morita K, Harao M (1989) Computation universality of one-dimensional reversible (injective) cellular automata, *Trans. IEICE, E-72*, 758-762.
17. Morita K, Ueno S (1992) Computation-universal models of two-dimensional 16-state reversible cellular automata, *IEICE Trans. Inf. & Syste., E75-D*, 141-147.
18. Pierzchala E, Perkowski M, Grygiel S (1994), A Field Programmable Analog Arrray for Continuous, Fuzzy and Multi-Valued Logic Applications," *Proc. ISMVL'94,* 148 - 155, Boston, MA.
19. Pierzchala E, Perkowski M (1999), Programmable Analog Array Circuit, U.S. Patent US5959871, Issued/Filed Dates: Sept 28 1999/Dec. 22, 1994.
20. Perkowski M, Pierzchala E (1993), New Canonical Forms for Four-Valued Logic, *Report, Electrical Engineering Department, PSU* . http://www.ee.pdx.edu/~mperkows
21. Perkowski M, Chrzanowska-Jeske M, Xu Y (1997) Lattice Diagrams Using Reed-Muller Logic. *Proc. of RM '97,* Oxford Univ., U.K., 85 - 102.
22. Perkowski M, Jozwiak L, Drechsle R, Falkowski B (1997). Ordered and Shared, Linearly Independent, Variable-Pair Decision Diagrams. *Proc. Intl. Conf. on Information, Communications and Signal Processing (ICICS'97),* Singapore.
23. Perkowski M, Pierzchala E, Drechsler R (1997), Layout-Driven Synthesis for Submicron Technology: Mapping Expansions to Regular Lattices, *Proc. 1st Int. Conf. on Information, Communications, and Signal Processing, ICICS'97,* Singapore.
24. Perkowski M, Pierzchala E, Drechsler R (1997), Ternary and Quaternary Lattice Diagrams for Linearly-Independent Logic, Multiple-Valued Logic and Analog Synthesis, *Proc. ISIC'97*, Singapore, Vol. 1, 269-273.

25. Perkowski M, Al-Rabadi A, Kerntopf P, Mishchenko A, Chrzanowska-Jeske M (2001), Three-Dimensional Realization of Multiple-Valued Functions using Reversible Logic, *Booklet ULSI'01*, 47-53, Warsaw, Poland.
26. Perkowski M, Al-Rabadi A, Kerntopf P, Buller A, Chrzanowska-Jeske M, Mishchenko A, Azad Khan M, Coppola A, Yanushkevich S, Shmerko V, Jozwiak L (2001) A General Decomposition for Reversible Logic, *Proceedings RM'2001, Starkville, Mississippi, USA, August 10-11, 2001*, 119-138.
27. Sasao T (1999) *Switching Theory for Logic Synthesis*, Boston: Kluwer Academic Publishers.
28. Shende VV, Prasad AK, Markov IL, Hayes JP (2002) Reversible Logic Circuit Synthesis, *Proceedings, 11th International Workshop on Logic Synthesis*, 125-130.