2001

Regularity and Symmetry as a Base for Efficient Realization of Reversible Logic Circuits

Marek Perkowski
Portland State University, marek.perkowski@pdx.edu

Pawel Kerntopf
Technical University of Warsaw

Andrzej Buller
ATR Kyoto, Japan

Malgorzata Chrzanowska-Jeske
Portland State University

Alan Mishchenko
Portland State University

See next page for additional authors

Citation Details

Perkowski, Marek; Kerntopf, Pawel; Buller, Andrzej; Chrzanowska-Jeske, Malgorzata; Mishchenko, Alan; Song, Xiaoyu; Al-Rabadi, Anas; Jozwiak, Lech; and Coppola, Alan, "Regularity and Symmetry as a Base for Efficient Realization of Reversible Logic Circuits" (2001). *Electrical and Computer Engineering Faculty Publications and Presentations*. 235.
https://pdxscholar.library.pdx.edu/ece_fac/235

Authors
Marek Perkowski, Pawel Kerntopf, Andrzej Buller, Malgorzata Chrzanowska-Jeske, Alan Mishchenko, Xiaoyu Song, Anas Al-Rabadi, Lech Jozwiak, and Alan Coppola

Regularity and Symmetry as a Base for Efficient Realization of Reversible Logic Circuits

Marek Perkowski, Pawel Kerntopf+, Andrzej Buller*, Malgorzata Chrzanowska-Jeske, Alan Mishchenko, Xiaoyu Song, Anas Al-Rabadi, Lech Jozwiak@, Alan Coppola$ and Bart Massey

PORTLAND QUANTUM LOGIC GROUP, Portland State University, Portland, Oregon 97207-0751.
*+Technical University of Warsaw, Poland, *ATR, Kyoto, Japan, @ Technical University of Eindhoven, $ Cypress Semiconductor Northwest*

Abstract

We introduce a ***Reversible Programmable Gate Array (RPGA)*** based on regular structure to realize binary functions in reversible logic. This structure, called a ***2 * 2 Net Structure***, allows for more efficient realization of symmetric functions than the methods shown by previous authors**.** In addition, it realizes many non-symmetric functions even without variable repetition. Our synthesis method to RPGAs allows to realize arbitrary symmetric function in a completely regular structure of reversible gates with smaller "garbage" than the previously presented papers. Because every Boolean function is symmetrizable by repeating input variables, our method is applicable to arbitrary multi-input, multi-output Boolean functions and realizes such arbitrary function in a circuit with a relatively small number of garbage gate outputs. The method can be also used in classical logic. Its advantages in terms of numbers of gates and inputs/outputs are especially seen for symmetric or incompletely specified functions with many outputs.

1. Introduction and Background

In [56] we introduced a method to realize symmetric and non-symmetric functions using 4*4 multi-valued Fredkin Gate of Picton. Picton himself used this gate for realization of Digital Summation Threshold Logic Device (DSTL) [34]. However, our previous design required two MV Fredkin gates per cell and introduced two garbage bits per cell, and Picton's approach used gates that are not realizable in truly reversible (quantum) logic. In both ours and Picton's approaches, all cells were programmed by the same constants. We found however that a better design can be done using the recently invented ***3 * 3 Kerntopf gate***. Our new design presented below has a regular structure and introduces only one garbage output bit per cell. Moreover, because the cell is controllable from the input, it can be also programmed to realize one more set of functions, which leads to essential improvement of the whole idea and to a new kind of programmable structure which we call RPGA. Not only is the cell able to create the AND/OR configuration used in symmetric and threshold circuits [56,34], but it allows also to obtain NAND/IMPLICATION configuration. As the result, smaller realizations of a wider class of symmetric functions can be found and less repetitions are required to realize arbitrary non-symmetric functions by variable repetitions [51,52].

As proved by Landauer [20],[19], it is a ***necessary*** condition for power not be dissipated in the circuit that the circuit be build from reversible gates (observe, *necessary but not sufficient*). *Reversible are circuits (gates) that have the same number of inputs and outputs and are one-to-one mappings between vectors of inputs and outputs; thus the vector of input states can be always uniquely reconstructed from the vector of output states*. *Conservative* are circuits that have the same number of ones in inputs and outputs. Our circuits are both reversible and conservative. Because truly low-power circuits cannot be built without the concepts of reversible logic, various technologies for reversible logic are recently intensively studied. These technologies include; 1) standard CMOS that can use university-available foundry, 2) optical technologies that can be realized with the state of the art materials, 3) quantum logic (QL) technologies, some of which proved to be physically realizable only very recently.

Reversible logic adders [6-12] and complete microprocessors [45] have been built, but using perhaps some "ad hoc" and not published logic design methods. Surprisingly little has been however published on systematic logic synthesis and optimization methods for reversible and quantum logic. In theory, classical logic synthesis methods can be used, but when adapted to reversible gates, they create unrealistically high numbers of additional gate output signals, making the circuit extremely complex. When applied to quantum logic, in addition they create an excessive number of Feynman gates used for wire crossing. A good synthesis algorithm for reversible logic (RL) should not create an excessive "garbage" [17] or "waste of outputs". Based on our previous research, we found that there is a very good "match" between the requirements of reversible logic [2,4,9,11,12,14,17,18,33,43,44] and the opportunities given by the regular logic/layout structure [46] and Linearly Independent Logic [47,48]. We believe that regular structures are good for reversible logic, because it is easier to re-use in them the additional outputs of reversible gates, instead of "wasting" them. In addition, we believe that the Linearly Independent Logic [46,47,48] with its reversibility properties should be useful as well. Observe for example, that the Shannon and Davio expansions are used for some outputs in fundamental reversible gates of Fredkin and Toffoli, and their generalizations can be used to create new multivalued and multi-input (more than 3) reversible gates.

As just one example of our general methodology based on these principles, we introduce here a <u>regular structure</u> to realize symmetric functions in binary reversible and quantum logic. This structure, although somehow similar to Lattice Structures introduced by us previously [50,51] and especially the MOPS structures [52] as well as to realizations from [28,34], is new and we call it a ***2 * 2 Net Structure.*** By a *regular structure* we understand a logic circuit and its physical layout structure being an array of identical cells regularly connected, or a structure composed of few, regularly connected, structures of this type, called *planes*. For instance, a well-known PLA is a regular structure with AND and OR planes. EXOR PLA is a regular structure with AND and EXOR planes. By *regularly connected*, we understand that every cell (except of boundary cells) is connected to its *k* neighbors. In the proposed structure, the cell, composed of two gates, has two inputs from neighbors, two outputs to neighbors, and two garbage outputs.

The method presented below allows to realize arbitrary symmetric function in a completely regular structure of reversible gates with little "garbage". By a *(totally) symmetric function* [49] we understand a Boolean function which is invariant to permuting any of its input variables. *Partially symmetric function* is invariant only to some input permutations. As discussed in [50] and the literature cited there, every Boolean function can be made symmetric, or *"is symmetrizable*". Functions are made symmetric by repeating their input variables, for instance a function F(a,b,c) that is not symmetric becomes symmetric when transformed to (incompletely specified) function F2(a1,b,a2,c) such that a1 = a2 =a and for every input vector (a,b,c) F(a,b,c) = F2(a1,b,a2,c). This function has an output don't care for every combination of inputs when a_1 != a_2. Because every (multi-output) Boolean function is symmetrizable, our method presented here is applicable to arbitrary multi-input, multi-output Boolean function. Recently efficient methods for symmetrization have been developed in our group, making application of the proposed here methods practical for at least some percent of non-symmetrical functions. These methods make use of more general definitions of symmetry than presented here [52]. We found that for high percent of practical function benchmarks the number of variable repetitions is small.

The goal of our Portland Quantum Logic research group is to create efficient logic synthesis methodologies for RL and QL, based on new gates, new structures (usually regular) and new design algorithms. The technique presented here is technology independent and can be thus used in association with any known or future reversible technology (also with different gates used in the structure). We believe, however, that the regularity of our networks will constitute an additional asset for the forthcoming technologies, especially nano-technologies.

2. Regular structure of Kerntopf and Feynman gates to realize an arbitrary symmetric function in RPGA.

2.1. Kerntopf and Feynman Gates

Kerntopf gate [18] is described by equations: ***P = 1 @ A @ B @ C @ AB, Q = 1 @ AB @ B @ C @ BC, R = 1 @ A @ B @ AC,*** where @ denotes ***EXOR.*** When ***C=1*** then ***P = A + B, Q = A * B, R = !C***, so ***MAX/MIN*** gate is realized on outputs ***P*** and ***Q*** with ***C*** as the controlling input value. When ***C = 0*** then ***P = !A * ! B, Q = A + !B, R = A @ B.*** Therefore for control input value 0 the gate realizes ***NAND*** and ***IMPLICATION*** on outputs ***P*** and ***Q.*** The ***Feynman gate***, called also ***controled-not*** or ***quantum XOR*** realizes functions ***P = A, Q = A @ B.*** When ***A = 0*** then ***Q = B,*** when ***A = 1*** then ***Q = !B***

2.2. Definitions

Let our area of interest be Boolean expressions that are specified as sum-of-products. There is a subset of such expressions in which every variable is either negated or not negated, but not both.

Definition 1. The variable that stands non-negated (positive) throughout the expression is called a ***positive polarity variable***. Variable that stands always in negative (negated) form (as a negated literal) is called a ***negative polarity variable***.
Note: If function has ***n*** variables, and each variable is either positive or negative, there are ***2^n*** different combinations of polarities of input variables, which are called ***polarities of the functional expression***. If all variables have positive polarity, the polarity of expression is called ***positive.*** If all variables have negative polarity, the polarity of expression is ***negative***. There are then ***2^n*** various ***polarities of expressions***.

Definition 2. *Unate function* is a function expressed only using AND and OR operators (for instance Sum-of-Products) in which every variable has either positive or negative polarity, but not both.
Example 1: function ***f1 = ab + b c'*** is unate and has polarity: a=positive, b=positive, c=negative. In short, it has the polarity (a,b,c) = (1,1,0), when positive polarity of variable is denoted by 1 and negative polarity by 0. Function f2 = a b' + a' b is not unate. This is EXOR gate, and it is a linear gate. All functions and gates can be characterized as: unate, linear and other. *Linear function* is an EXOR operator of literals and constants. Function majority of three variables: f3 = ab + ac + bc is both symmetric and positive unate. Functions that are both positive unate and totally symmetric will be of our interest in this paper.

Definition 3. Totally Symmetric function that has value 1 when exactly ***k*** of its ***n*** inputs are equal one and exactly ***n-k*** remaining inputs are equal 0, is called a ***single-index symmetric function,*** denoted by ***$S^k(x_1, x_2, \ldots, x_n)$.*** Analogously, by ***$S^{\{i,j,k\}}$*** we denote the function that is one when I, j, or k of its variables are equal one. Obviously, this notation can be extended to any number of indices, so every symmetric function can be written as $S^{I}(x_1, x_2, \ldots, x_n)$, where I is any subset of the set of indices $\{0,1, 2,\ldots n\}$. It can be also easily checked that:

$$S^{I1}(x_1, x_2, \ldots, x_n) \text{ AND } S^{I2}(x_1, x_2, \ldots, x_n) = S^{I1 \text{ intersection } I2}(x_1, x_2, \ldots, x_n) \quad (1)$$

$$S^{I1}(x_1, x_2, \ldots, x_n) \text{ OR } S^{I2}(x_1, x_2, \ldots, x_n) = S^{I1 \text{ union } I2}(x_1, x_2, \ldots, x_n) \quad (2)$$

$$S^{I1}(x_1, x_2, \ldots, x_n) \text{ XOR } S^{I2}(x_1, x_2, \ldots, x_n) = S^{I1 \text{ symmetric\_difference } I2}(x_1, x_2, \ldots, x_n) \quad (3)$$

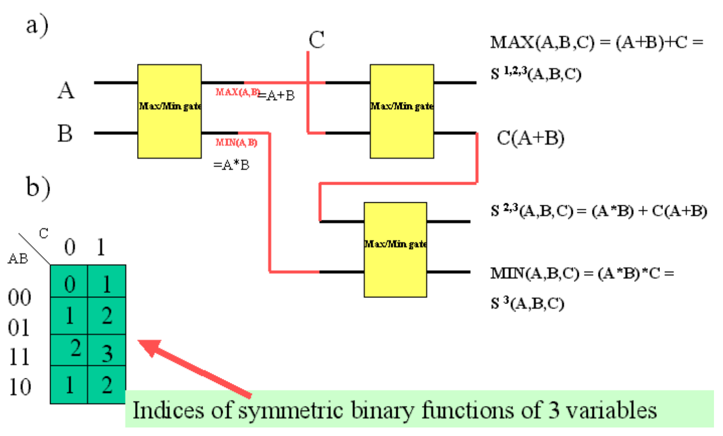

Figure 1. Example of realization of some symmetric functions of three variables in the left plane from Fig. 2: (a) regular structure from reversible MAX/MIN gates that realizes positive polarity unate symmetric functions, (b) indices of a symmetric function of variables A,B,C; each cell includes an index of a symmetric function corresponding to it, for instance, function $S^{2,3}(A,B,C)$ will have ones in cells with indices 2 and 3 and zeros in cells with indices 0 and 1.

3.2. 2* 2 Net Structures and RPGAs

Our regular structure of RPGA has two regular planes (Fig. 2). The first plane from left is a levelized triangular structure in which the input variables correspond to the columns (we will call it also the ***triangular plane***). The programming of part of the first plane to OR/AND (MAX/MIN) combination is shown in Figure 1. In contrast to Lattices, however, the structure from Figure 1, when realizing arbitrary multi-output function with geometrically adjacent output signals does not require variable repetition. (For some multi-output functions such as "counter of ones in a binary string" the Lattice without repeated variables required to push the output signals by few spaces apart. Or, these output signals were adjacent but the input variables had to be repeated. This was a disadvantage of Lattices for large multi-output symmetric functions, which was next attempted to improve in MOPS circuits [52]). The structure of the first plane is planar, regular and algorithmically created. It realizes all ***positive unate symmetric functions (PUS)*** of its input variables. The second structure from left in Figure 2 is just a sequence of columns of Feynman gates that converts these PUS functions to arbitrary symmetric functions. A plane of Feynman gates uses its internal EXOR gates to realize every output function as an EXOR of PUS functions from plane two. (Formula **(3)** above is used). This plane can be compared in its functionality to the collecting OR plane in a standard AND/OR PLA that is used to realize a Sum-of-Product (SOP, DNF) expression. It uses , however, EXOR as the collecting gate, similarly as in ESOP PLAs. Figure 1 illustrates how positive polarity unate symmetric functions can be created systematically in a regular planar arrangement of MAX/MIN modules. Observe that each vertical output function, from top to bottom, includes the next function and are all positive polarity and unate. The sets of indices of the adjacent functions differ by one. Let us observe that positive unate symmetric functions generated on the outputs of the triangular plane have a very nice property (3). However, because the EXOR gate is not reversible, we have to complete it to Feynman gate by repeating one of its inputs to the output. Because our structure is regular, this does not complicate the structure. In result, we obtain a structure such as shown in Figure 2. As we see, in this regular structure, we obtain not only the symmetric functions of all variables, but also some additional functions. In addition to the method from this section, we created a method to use ***all these functions*** to synthesize arbitrary multi-output functions by EXOR-ing them in Feynman gates in the collecting plane.

4. Observations on notation

It is very important for logic synthesis using reversible/quantum gates to have a good notation. Currently there are three notations for reversible/quantum circuits: ***Picton's notation*** [34] shows only the standard inputs and outputs, as illustrated in Figure 4. The disadvantage of this notation is that it does not emphasize the cost of the programming (constant) inputs and the garbage outputs, so it is not a good heuristic to design quantum circuits, for instance the property of no wire crossing is not clearly seen. ***Fredkin's notation*** [17] shows also the individual gates in arbitrary (not timed) layout, but it uses quantum notation for gates and shows all inputs and outputs of gates, so that the garbage is clearly seen in the schematics (Fig. 2).

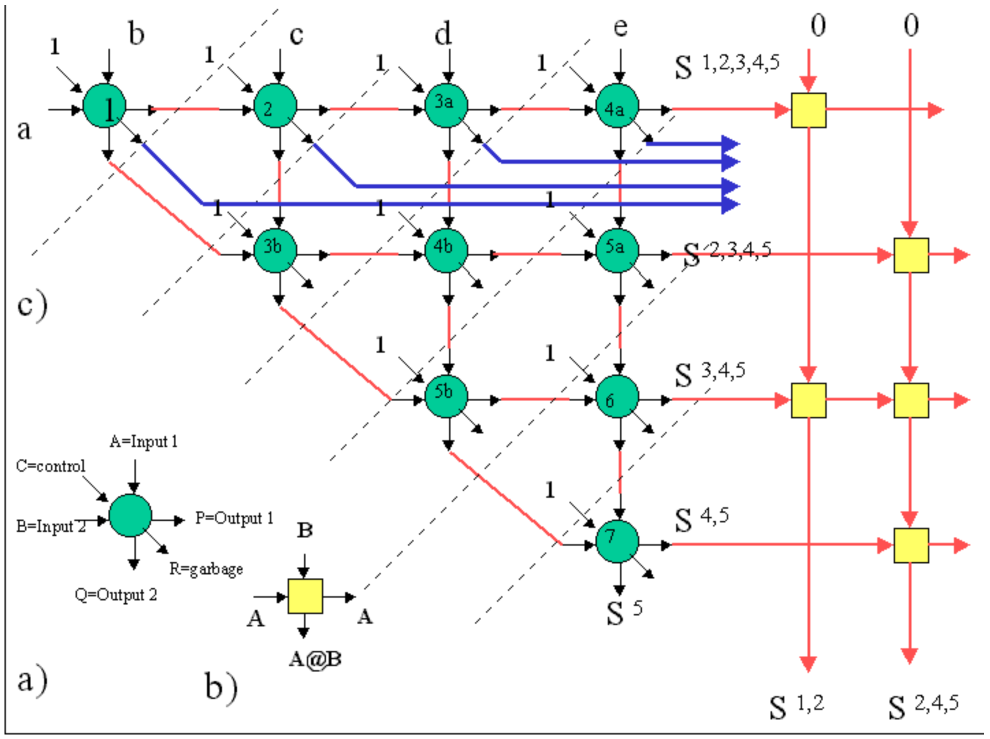

Figure 2. Using Fredkin's notation to draw the structure of RPGA: (a) the notation for a Kerntopf Gate used as a cell in first planel, (b) the notation for a Feynman gate used in the collecting plane, (c) realization of 5-input 2-output function $<S^{1,2}(a,b,c,d,e)\ ,\ S^{2,4,5}(a,b,c,d,e)>$ *in RPGA.*

Observe in Figure 2 that horizontal outputs from Kerntopf gates PUS functions which are EXOR-ed using Feynman gates in the right plane to create arbitrary symmetric functions at the bottom. Additional garbage outputs of Kerntopf gates which in any case must be forwarded to the primary output (shown in bold for cells in upper row only) can be the inputs to Feynman gates in the same way as the horizontal outputs, which extends the class of realizable functions in the structure with no repeated variables. The interrupted lines delineate levels of gates – this is useful to draw the Feynman-like quantum diagram and in timing analysis. Observe that in this diagram not only garbage outputs must be extended to the primary circuit's outputs but also constant control inputs of gates should be taken from primary constant inputs of the circuit. This would make the circuit visually more complicated but this is the only correct quantum interpretation. This figure clearly shows that reducing (unavoidable) garbage is the main design problem of reversible computing from the logic synthesis and physical design point of view. Integrated "layout-driven logic synthesis tools" should be thus created.

Conclusion. We showed the practical advantage of Kerntopf gate over classical Fredkin and MV Fredkin gate of Picton to realize symmetric functions. Although the number of wires in our circuit may look excessive at the first glance to people unfamiliar with reversible logic, its comparisons with realizations obtained using other methods [6-12,17,31-33] illustrate the true advantages of using regular structures of this type. The known methods create truly excessive numbers of "garbage" outputs for larger benchmarks. Moreover, the circuit shown and the synthesis method for it can be significantly further improved, we have no space here to present these improvements. Because of novelty of this topic we give full list of references. It can be shortened, if necessary.

Literature

[1] W.C. Athas & L."J." Svensson , "Reversible Logic Issues in Adiabatic CMOS", Exploratory Design Group, University of Southern California - Information Sciences Institute, Marina del Rey, CA 90292-6695, {athas,svensson}@isi.edu
[2] Ch.H. Bennett , "Notes on the History of Reversible Computation", *IBM J. Res. Develop.,* Vol. 32, 1988, pp. 16-23.
[3] Ch. H. Bennett and R. Landauer, "The Fundamental Limits of Computation", *Scientific American,* July 1985, pp. 38-46.
[4] Bennett, C., "Logical reversibility of computation", *I.B.M. Journal of Research and Development*, **17** (1973), pp. 525-532.
[6] A. De Vos, "Proposal for an Implementation of Reversible Gates in c-MOS", *Int. Journal of Electronics*, Vol. 76, 1994, pp. 293-302.

[7] A. De Vos, "Reversible Computing in c-MOS", *Proc. Advanced Training Course on Mixed Design of VLSI Circuits,* 1994, pp. 36-41. [8] A. De Vos, "A 12-Transistor c-MOS Building-Block for Reversible Computers*", Int. Journal of Electronics,* Vol. 79, 1995, pp. 171-182. [9] A. De Vos, "Reversible and Endoreversible Computing", *Int. Journal of Theor. Phys.,* Vol. 34, 1995, pp. 2251-2266. [10] De Vos, A.: "Introduction to r-MOS systems"; *Proc. 4 th Workshop on Physics and Computation,* Boston, 1996, pp. 92-96. [11] De Vos, A.: "Towards reversible digital computers"; *Proc. European Conference on Circuit Theory and Design,* Budapest, 1997, pp. 923-931. [12] De Vos, A.: "Reversible computing"; *Progress in Quantum Electronics*, **23** (1999), pp. 1-49.
[13] B. Desoete, A. De Vos, M. Sibinski, T. Widerski, "Feynman's Reversible Logic Gates Implemented in Silicon", *Proc. 6 th Intern. Conf. MIXDES*, 1999, pp. 497-502. [14] R.Feynman, "Quantum Mechanical Computers", *Optics News,* **11** (1985), pp. 11-20. [15] R. Feynman, "There's plenty of space at the bottom: an invitation To Enter a New Field of Physics," *Nanotechnology,* Ed BC Crandal and J.Lewis, the MIT Press 1992, pp. 347-363 [16] Feynman, R.: "Feynman lectures on computation" (A. Hey and R. Allen, eds); *Addison-Wesley,* Reading (1996). [17] E. Fredkin, T. Toffoli, "Conservative Logic", *Int. Journal of Theor. Phys.,* **21** (1982), pp. 219-253. [18] P. Kerntopf, "Logic Synthesis Using Reversible Gates," *Proc. 3rd Symposium on Logic, Design and Learning,* Portland, Oregon, May 31, 2000. P. Kerntopf, "A Comparison of Logical Efficiency of Reversible and Conventional Gates," *9th IEEE Workshop on Logic Synthesis*, P. Kerntopf, "On Efficiency of Reversible Logic (3,3) – Gates. *" Proc. 7th Intl. Conf. MIXDES, 2000*, pp. 185-190. [19] R. Keyes, and R. Landauer, "Minimal energy dissipation in logic"; *I.B.M. Journal of Research and Development,* **14** (1970), pp. 153-157. [20] R. Landauer, "Irreversibility and heat generation in the computational process"; *I.B.M. Journal of Research and Development,* **5** (1961), pp. 183-191. [21] J. Lim, D. Kim, and S. Chae, "Reversible Energy Recovery Logic Circuits and Its
8-Phase Clocked Power Generator for Ultra-Low-Power Applications," IEICE Trans. Electron, OL.E82 –C, No. 4 April 1999. [23] N. Margolus, "Physics and Computation", *Ph. D. Thesis*, Massachussets Institute of Technology, USA 1988. [24] L.J. Micheel, A.H. Taddiken and A.C. Seabaugh, "Multiple-Valued Logic Computation Using Micro- and Nanoelectronic Devices," *Proc. ISMVL,* IEEE, 1993, pp. 164-169 [25] R C. Merkle and K. Eric Drexler; "Helical Logic", WWW. [26] R C. Merkle, "Reversible electronic logic using switches," *Nanotechnology,* **4** (1993) pp. 21-40 [27] R. C. Merkle, "Two types of mechanical reversible logic," *Nanotechnology,* **4** (1993), pp. 114-131. [28] O.N. Muzychenko, "Uniform and Regular Structures for Realization of Symmetric Functions of the Algebra of Logic", Automation and Remote Control, Vol. 59, No.4, 1998, pp. 581-592 [29] Y.N. Patt , "A Complex Logic Module for the Synthesis of Combinational Switching Circuits", *Proc. 30 th AFIPS Spring Joint Computer Conf.,* 1967, pp. 699-706. [30] A. Peres, "Reversible Logic and Quantum Computers", *Physical Review A,* **32** (1985), pp. 3266-3276. [31] P. Picton, "Optoelectronic, Multivalued, Conservative Logic", *Int. Journal of Optical Computing,* Vol. 2, 1991, pp. 19-29. [32] P. Picton, "Multi-valued Sequential Logic Design Using Fredkin Gates", *MVL Journal,*
vol.1, 1996, pp.241-251. [33], P. Picton, "A Universal Architecture for Multiple-valued Reversible Logic," *MVL Journal,* **5** (2000), pp.27-37. [34] P. Picton, "Modified Fredkin Gates in Logic Design," *Microelectronics Journal*,
25 (1994), pp. 437-441.[35] M.R. Rayner, D.J. Newton, "On the Symmetry of Logic ", *Journal of Physics A: Mathematical and General,* **28** (1995), pp. 5623-5631. [36] G. Stix, "Riding the back of electrons"; *Scientific American*, 279 (September 1998), pp. 20-21. [37] J. Shamir, H. J. Caulfield, W. Micelli, W., and R.I. Seymour, "Optical Computing and the Fredkin Gates", *Applied Optics,* **25,** pp. 1604-1607, 1986. [38] J.A. Smolin, and D.P. DiVincenzo, "Five Two-Bit Quantum Gates are sufficient to Implement the Quantum Fredkin Gate", *Physical Review A*, 53, pp. 2855-2856.
[39] L. Storme, A. De Vos, G. Jacobs, "Group Theoretical Aspects of Reversible Logic Gates", *Journal of Universal Computer Science,* Vol. 5, 1999, pp. 307-321.
[40] T. Toffoli, "Reversible Computing", in Automata, Languages and Programming, *Springer Verlag,* 1980, pp. 632- 644.
[41] P. Wayner, " Silicon in Reverse", *Byte Magazine*, August 1994, page 67.
[42] A workshop on the Physics of Computation was held at MIT in 1981; the papers were printed in the April, June and December issues of the 1982 International Journal for Theoretical Physics, Volume 21.
[43] V. I. Varshavsky, "Logic Design and Quantum Challenge". *Preprint from the author.*
[44] J. Birnbaum, "Computing Alternatives", Talk given on March 3, 1997 atACM97, March 3, 1997, San Jose, California, by Director of Hewlett-Packard Laboratories, Senior Vice-President of Research and Development.
[45] M. Frank, "Physical Limits of Computing," CIS 4930.1194X / 6930.1078X, Spr. '00. http://www.cise.ufl.edu/~mpf/course.html
[46] A. Sarabi, N. Song, M. Chrzanowska-Jeske, M. A. Perkowski, "A Comprehensive Approach to Logic Synthesis and Physical Design for Two-Dimensional Logic Arrays," *Proc. DAC'94,* San Diego, June 1994, pp. 321 - 326.
[47] M. A. Perkowski, "A Fundamental Theorem for EXOR Circuits," *Proc. of IFIP W.G. 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design,"* Hamburg, Germany, September 16-17, pp. 52 - 60, 1993.
[48] M. Perkowski, B. Falkowski, M. Chrzanowska-Jeske, and R. Drechlser, ``Efficient Algorithms for Creation of Linearly-Independent Decision Diagrams and their Mapping to Regular Layouts". *VLSI Design*. In print
[49] Z. Kohavi, "Switching Functions and Finite Automata Theory, *Prentice Hall.*
[50] E. Pierzchala, M. A. Perkowski, S. Grygiel, "A Field Programmable Analog Arrray for Continuous, Fuzzy and Multi-Valued Logic Applications," *Proc. ISMVL'94,* pp. 148 - 155, Boston, MA, May 25-27, 1994.
[51] M. Chrzanowska-Jeske, Y. Xu, and M. Perkowski, ``Logic Synthesis for a Regular Layout," VLSI Design, Vol. 10, No. 1, pp. 35 - 55, 1999.
[52] M.A. Perkowski, M. Chrzanowska-Jeske, and Y. Xu, ``Multi-Level Programmable Arrays for Sub-Micron Technology based on Symmetries," *Proc. ICCIMA'98 Conference,* pp. 707-720, February 1998, Australia, published by *World Scientific.*
[55] M.A. Perkowski. A. Al-Rabadi, P. Kerntopf, M. Chrzanowska-Jeske and A.Mishchenko, "Three Dimensional Realization of Multi-Valued Symmetric Functions using Reversible Logic". Submitted to ULSI 2001.
[56] 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", submitted to Euro-Micro 2001.
[57] M.A. Nielsen and I.L. Chuang, "Quantum Computation and Quantum Information", Cambridge, 2001.