From Massively Parallel Image Processors to Fault-Tolerant Nanocomputers

Jie Han and Pieter Jonker
Quantitative Imaging Group, Faculty of Applied Sciences, Delft University of Technology
Lorentzweg 1, 2628 CJ, Delft, The Netherlands
Emails: jie@ph.tn.tudelft.nl, pieter@ph.tn.tudelft.nl

***Abstract*— Parallel processors such as SIMD computers have been successfully used in various areas of high performance image and data processing. Due to their characteristics of highly regular structures and mainly local interconnections, SIMD or SIMD-like architectures have been proposed for a large-scale integration of recently developed quantum and nanoelectronic devices. In this paper, we present a fault-tolerant technique suitable for an implementation in nanoelectronics, the triplicated interwoven redundancy (TIR). The TIR is a general class of triple modular redundancy (TMR), but implemented with random interconnections. A prototype structure for an image processor is proposed for the implementation of the TIR technique and a simulation based reliability model is used to investigate its fault-tolerance. The TIR is extended to higher orders, namely, the N-tuple interwoven redundancy (NIR), to achieve higher system reliabilities. It is shown that the reliability of a general TIR circuit is, in most cases, comparable with that of an equivalent TMR circuit, and that the design and implementation of restorative devices (voters) are important for the NIR (TIR) structure. Our study indicates that the NIR (TIR) is in particular suitable for an implementation by the manufacturing process of stochastically molecular assembly, and that it may be an effective fault-tolerant technique for a massively parallel architecture based on molecular or nanoelectronic devices.**

I. Introduction

Historically, parallel processing offered considerable performance advantages in many areas of computing. Several approaches have been explored and prototype systems were constructed. Among those, SIMD (single instruction and multiple data) computers have been successfully used in various areas of data processing [1], [2]. The field of high performance image processing in particular has brought forward archetypical systems (see, for examples, [3]-[6]). This evolvement of massively parallel processors have been based on the continuous miniaturization of electronic components. As today's CMOS technology enters the nanoelectronic realm (tens of nanometers and below), where quantum mechanical effects start to prevail, conventional CMOS devices are meeting many technological challenges for further scaling [7]. Novel information processing devices based on new physical phenomena have been investigated, and various novel architectures have been proposed for large-scale integration of these devices [8]. Recent progress in molecular electronics has in particular motivated much effort in the research of architectures that are suitable for the implementation of a nanoelectronic computer [9]-[11].

Due to the characteristics of many nanoelectronic devices, such as low power consumption, low drive capability and easy local interactions, parallel architectures that are highly regular and locally connected, have been studied as candidate architectures for nanocomputers. The architectural issues of a SIMD array have been discussed in [12] for nanoscale devices. Since molecular circuits are likely to be assembled through a bottom-up manufacturing process, in which randomness is inherent, imprecisions and inaccuracies seem to be inevitable in future nanoelectronic systems. Fault tolerance is thus a major issue in nanoarchitecture design [13]. Conventional fault-tolerant techniques, such as NAND multiplexing [14], N-tuple modular redundancy (e.g. triple modular redundancy (TMR)) [15] and reconfiguration, have been investigated for implementations in nanoelectronic systems [16]-[22]. A hierarchically reconfigurable architecture with the multiplexing technique implemented into the fundamental circuits has been studied for a system robust against both manufacturing defects and transient faults [23].

Von Neumann's multiplexing technique is based on a massive duplication of imperfect devices and randomized imperfect interconnects. It has been shown that this construction can be reliable with a high probability, provided that the failure probability of a component is sufficiently small. As implied in the multiplexing technique, NMR and TMR designs have been used as benchmarks for evaluating fault-tolerant approaches and were implemented in VLSI for high reliability applications [15]. NMR techniques, generally implemented at modular level instead of gate level, use redundant components to mask the effect of faults. In TMR, the most general form of NMR, three identical modules perform the same operation, and a voter accepts outputs from all three modules, producing a majority vote at its output. This majority voter functions as a restoring organ, bringing the outputs to a more reliable level.

In this paper, we present a fault-tolerant technique suitable for an implementation by the manufacturing process of stochastically molecular assembly, the N-tuple interwoven redundancy (NIR) (in particular, the triplicated interwoven redundancy (TIR)). The TIR is presented as a general class of triple modular redundancy (TMR), but implemented with random interconnections. A simulation based reliability model is used to investigate the fault-tolerance of the TIR implementations of a 1-bit processor structure. The TIR is extended to higher orders of NIR, to achieve higher system reliabilities.

Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)
1051-4651/04 $ 20.00 IEEE

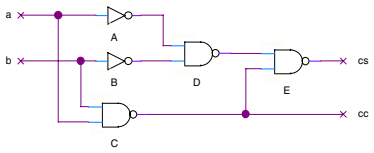

Fig. 1. A nonredundant complementary half adder implemented with NAND logic

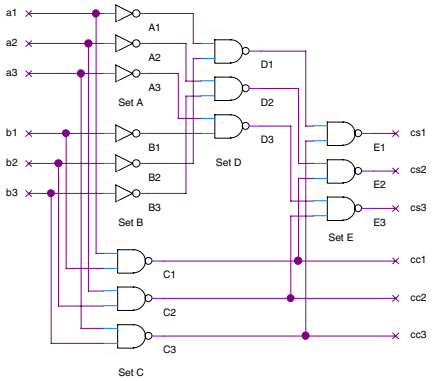

Fig. 2. A TIR implementation of the complementary half adder

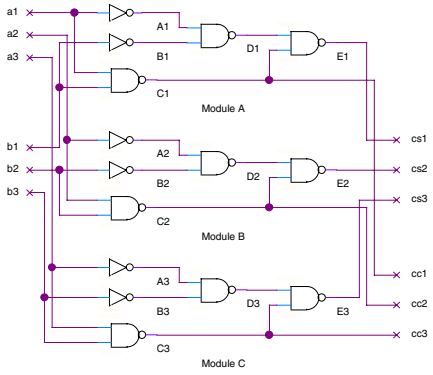

Fig. 3. The TMR configuration of the TIR complementary half adder

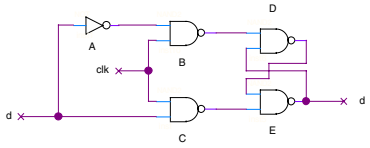

Fig. 4. A clocked D-type flip-flop implemented with NAND logic

II. Triplicated Interwoven Redundancy (TIR)

The idea of triplicated interwoven redundancy (TIR) originates from von Neumann's multiplexing technique and the general concept of the interwoven redundant logic [24]. To illustrate it, we show the schematics of a complementary half adder (computing the complements of carry and sum, denoted as cc and cs) in Fig. 1 and a TIR implementation of it in Fig. 2, both implemented with NAND gates (including inverters). In the TIR complementary half adder each NAND gate in the nonredundant circuit is replaced by a triplication and all the interconnections are accordingly triplicated as well. A TIR circuit thus has three times as many gates and interconnections as the corresponding nonredundant circuit.

The interconnections in a TIR circuit are in principle arranged in a random way. In practical implementation the random interconnections can be substituted by arbitrarily selected static ones that have specific routes. In a TIR circuit made up of 2-input NAND gates, for instance, there are six possible pair connections $\{(1,1),(2,2),(3,3)\}$, $\{(1,1),(2,3),(3,2)\}$, $\{(1,2),(2,3),(3,1)\}$, $\{(1,2),(2,1),(3,3)\}$, $\{(1,3),(2,1),(3,2)\}$ and $\{(1,3),(2,2),(3,1)\}$ (by (i,j) we mean here the output of gate i in a triplication is paired with the output of gate j in another triplication to form the inputs of a gate in the next stage). The total interconnect patterns become 36 (6×6) if a distinction is made among the gate orders of a triplication in the next stage. One method to arrange the interconnections is to randomly adopt one of the 36 connection patterns for all connecting pairs in adjacent layers. As shown in Fig. 2, the interconnect patterns used in the three layers from inputs to outputs of the circuit are $\{(1,1),(2,2),(3,3)\}$, $\{(1,2),(2,3),(3,1)\}$ and $\{(1,3),(2,1),(3,2)\}$, while the circuit can be with any other interconnect patterns.

It is interesting to notice that, if the pattern $\{(1,1),(2,2),(3,3)\}$ is used in all layers for all interconnections, the three modules in Fig. 2 will independently perform computation, actually working as a TMR circuit, as depicted in Fig. 3. The TIR is hence a general class of TMR implemented with random interconnections and the TMR is a particular configuration of TIR with regular interconnections. The randomness in the interconnects of TIR is particularly interesting for the integration of molecular electronics, for which stochastically chemical assembly is most likely to be the manufacturing method.

The principle of TIR is also applicable to the circuits of storing devices. In Fig. 4 a clocked D-type flip-flop is drawn and a TIR implementation is shown in Fig. 5.

Similarly as in TMR, a decision element is needed in a general TIR circuit as a restoring device. A design of a 2-level majority voter, consisting of four NAND gates, is shown in Fig. 6. This voter is favorable for applications based on the transistors. In quantum and nanoelectronic regime, however, the implementation of majority logic could be greatly simplified. A simple structure of a single majority gate, as schematically shown in Fig. 7, has been proposed for possible implementations in nanoelectronic circuits [25]-[27].

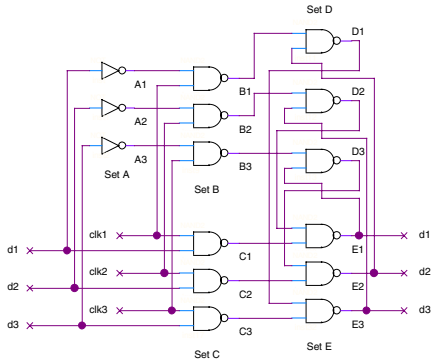

Fig. 5. A TIR implementation of the clocked flip-flop

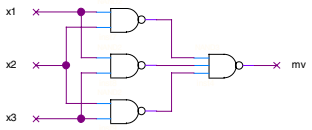

Fig. 6. A 2-level majority voter

III. A Processor Prototype for Array Architectures

A typical processor structure for a processor array consists of an arithmetic and logic unit (ALU), registers and a multiplexer, as shown in Fig. 8. Upon the arrival of clock signals, the input signals are latched in the registers and a computation procedure is triggered. At the end of the computation, results are sent to memory and neighbors. This processor prototype has been widely used to model processor arrays, and many variations based upon it have been successfully implemented in parallel processors for high-performance image processing and pattern recognition (see, for example, [3]-[6]).

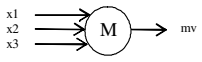

Fig. 7. A single majority gate

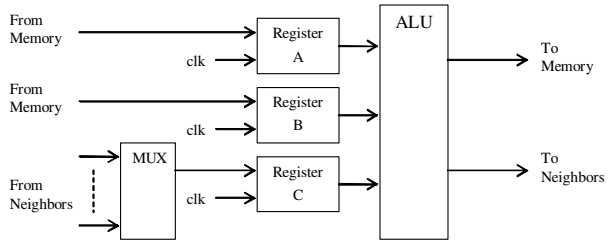

Fig. 8. A prototype structure of PE for an image processor array

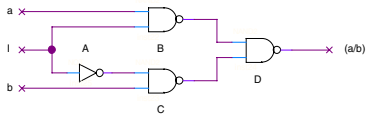

Fig. 9. A 2-to-1 multiplexer

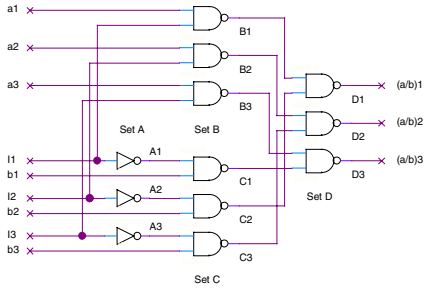

Fig. 10. A TIR implementation of the 2-to-1 multiplexer

In a 1-bit processor, the ALU is basically composed of a full adder, which can be constructed from the circuits of the complimentary half adder in Fig. 1 plus a few auxiliary gates. The registers, used to store the inputs from memories and neighboring processors, can be realized by the clocked flip-flops shown in Fig. 4. The fault-tolerant implementations of the ALU and registers can thus be obtained from the TIR circuits of the complementary half adder and flip-flop discussed in the previous section.

A 2-to-1 multiplexer requiring one instruction bit I is shown in Fig. 9. For simplicity, we only show the schematic of a TIR implementation of the 2-to-1 multiplexer in Fig. 10, which is sufficient for a linear processor array [28].

Hence, all the elements composing the processor structure in Fig. 8 are available in their fault-tolerant implementations of TIR circuits. In the next section a fault injection simulation is proposed to study the reliability of these fault-tolerant processor structures.

IV. A Simulation Based Reliability Model and Analysis

In this study, a fault injection simulation procedure has been performed to investigate the effects of multiple component failures in TIR structures. It goes as follows:

1) Start at all good states. A number of faulty gates (the number, m, starting from 0 and up to the maximum number of faulty gates in the circuit) are randomly selected from all of the gates in the fault-tolerant circuit and a randomly selected stuck-at-0 or stuck-at-1 fault is emulated at the output of each faulty gate.
2) A set of input pattern is applied to the fault-tolerant circuit, in which faults have been injected. The outputs produced in the circuit are then compared with the

correct ones. If the fault-tolerant circuit provides the correct outputs, repeat Step 2 until the complete set of input patterns have been tested. Otherwise, increase the number of failed simulations, k_m, by 1.

3) Increase the number of simulations performed thus far, n, by 1. If n is less than the total number of simulations to be performed, N, go to Step 1. Otherwise, go to Step 4.
4) Compute the failure rate of the simulated fault-tolerant circuit by the number of injected faults, $F_m = k_m/N$. Increase the number of faults injected into the fault-tolerant circuit, m, by 1. If m is no larger than the maximum number of faulty gates in the fault-tolerant circuit, go to Step 1. Otherwise, end the simulation.

In the simulation, faults were considered to appear in logic gates, and were injected at once before the circuit is put into operation. This is likely to be the case for manufacturing defects. A simulation procedure was proposed in [29] for the modeling of transient faults that spontaneously appear during system operation.

At the end of each simulation, a failure rate by the number of faults injected into the fault-tolerant circuit, F_m, is obtained as the number of times the fault-tolerant circuit fails, k_m, divided by the total number of fault injection simulations performed, N.

The probability distribution of the number of random faults in a fault-tolerant circuit follows the binomial distribution, i.e.

$$P(m) = \binom{N}{m} p_f^m (1 - p_f)^{N-m}, \quad (1)$$

where p_f is the error rate of a logic gate in the circuit, m and N are respectively the number of faulty gates and the total number of gates in the circuit.

By considering the number of faults, m, to be a random variable, the failure rate F_m can be used as a failure distribution to m. The reliability of the fault-tolerant circuit is therefore obtained by summing over all conditional reliabilities with the presence of faults, i.e.

$$R_{FT} = \sum_{m=0}^{N} \binom{N}{m} (1 - k_m/N) p_f^m (1 - p_f)^{N-m}. \quad (2)$$

Hence, the reliability of a fault-tolerant circuit can be obtained from the simulation based formula (2).

In the simulation the random interconnections in TIR circuits are substituted by randomly selected static ones, of which TMR is a specific configuration with regular interconnections. We have simulated the TIR processor structures with various interconnect patterns (including the one of TMR). The reliabilities obtained from the simulation based formula (2) are plotted against the error rate of a NAND gate in Fig. 11 for six sets of different interconnect patterns.

As revealed in the figure, the TIR structures with randomly selected interconnections present slightly lower probabilities of system survival than the TMR structure. In most cases, however, these variances in reliability are hardly discernable.

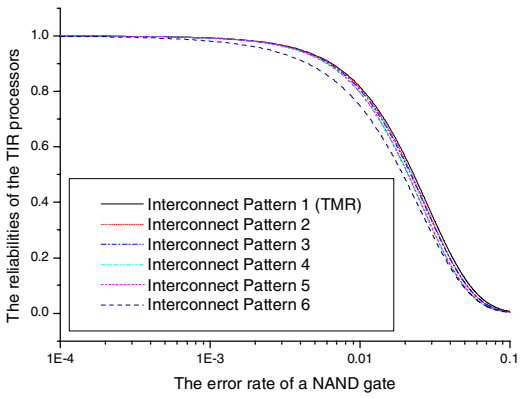

Fig. 11. The reliabilities of TIR circuits with random interconnects (including the TMR with regular interconnects)

For a small fraction of interconnect patterns (e.g. interconnect pattern 6 in Fig. 11), this variance becomes noticeable. Our further investigations show that this is due to a fanout effect of erroneous signals and a malicious interconnect combination of the output gates in the flip-flop (gates D and E in Fig. 5) of Register C (in Fig. 8). As a result, any single error in the multiplexer (MUX in Fig. 8), if propagated into the flip-flop through its fanout circuit, causes failures of two outputs of the flip-flop. These failures present a majority and thus cannot be corrected by the voters attached to the circuit.

Fortunately, there is just one of 36 possible interconnect combinations of the output gates that would cause the failure of the flip-flop (and thus the processor) due to a single error in the multiplexer. Generally, the variances in reliability obtained from various configurations of TIR (including TMR) are small so that they are negligible. Virtually, these TIR circuits are equivalent in terms of reliability without a distinction in interconnect patterns. This randomness in interconnects is a prominent feature of TIR, though in some cases it may introduce a decrease in reliability improvements.

V. N-tuple Interwoven Redundancy (NIR)

The TIR, as a general class of TMR, can readily be extended to, say, N-tuple interwoven redundancy (NIR), similarly as TMR to NMR. Thus, the NIR is a general class of NMR, but with random interconnections.

The degree of redundancy R used to construct an NMR system is determined from the desired number of faulty circuit modules to be masked, E, by [30]

$$2E + 1 \le R \le (E + 1)^2. \quad (3)$$

A grouping parameter, K, is used to design the voter circuit of an NMR. For a 2-level voter circuit, K indicates the number of inputs of a gate in the first level of the voter [30], and

$$E + 1 \le K \le R - E. \quad (4)$$

For TMR and the 2-level voter circuit in Fig. 6, for instance, we have $E = 1$, $R = 3$ and $K = 2$. For a general NMR system, the number of gates in the first level of a voter can be obtained by selecting K-out-of-R elements, and is given by

$$c = \binom{R}{K} = \frac{R!}{K!(R-K)!}. \quad (5)$$

In practice, applications of NMR systems are mainly restricted to odd numbers of replications, i.e. $R = 3, 5, 7, 9, ...$

To investigate the fault-tolerance of NIR systems, we have studied the NIR implementations of the processor structure in Fig. 8, with $N = 3$ (i.e. TIR), $5, 7$ and 9. Similarly as in TIR, an NIR voter circuit can follow a design of the 2-level structure in Fig. 6 or a design of the single majority gate structure in Fig. 7. For the 2-level voter design, the numbers of NAND gates in NIR voters for $R = 3, 5, 7$ and 9 ($E = 1, 2, 3$ and 4) are $4, 11, 36$ and 127, according to equation (5). We can see that the size of a voter grows faster than the increase of the redundancy in an NIR circuit. This implies that the performance gain by a higher degree of redundancy may be degraded by an increased complexity of an NIR voter. For a single majority gate design, however, the complexity of an NIR voter is kept similar or slightly increased due to an increase in gate interconnections in a higher order of NIR.

The reliabilities obtained by simulations are plotted against the gate error rate in Fig. 12, for the NIR circuits using these two types of voters. As can be seen, indeed, the use of a higher order of redundancy ($R = 5, 7$ or 9) does not offer a better fault-tolerance if a 2-level voter design is used in an NIR system. In fact, the system degenerates to a level that is less reliable than the nonredundant one. The higher the degree of redundancy is, the worse the reliability. If a single majority gate is used as a voter, however, a higher reliability results in an NIR system with a higher degree of redundancy, i.e., an improved system reliability is obtained by using a higher order of NIR. These indicate the significance of voter implementations in an NIR system.

With the use of single majority voters, a reliability improvement can only be obtained when the gate error rate in an NIR system is lower than a threshold. As revealed in Fig. 12, this threshold has a higher value in a higher-order NIR system. For instance, a TIR system ($R = 3$) does not afford an advantage over the nonredundant structure until the error rate of a gate reaches approximately 0.02, while an equivalent NIR structure with $R = 9$ provides a better performance as long as the gate error rate is not larger than approximately 0.05.

VI. Conclusion

Inspired by the success of massively parallel processors in the field of high performance image processing, we have explored the possibility of building a fault-tolerant processor structure with unreliable nanoelectronic devices. The triplicated interwoven redundancy (TIR) is presented as a general class of triple modular redundancy (TMR), but implemented with random interconnections. The TIR implementations of a 1-bit processor element are studied by using a simulation-based reliability model. The TIR is extended to higher orders, namely, the N-tuple interwoven redundancy (NIR), in order to achieve higher system reliabilities. In general, the reliability of a TIR circuit is comparable with that of an equivalent TMR circuit, while for certain interconnect patterns the TIR structure may present an inferior performance to TMR, due to its interwoven nature in gate interconnections.

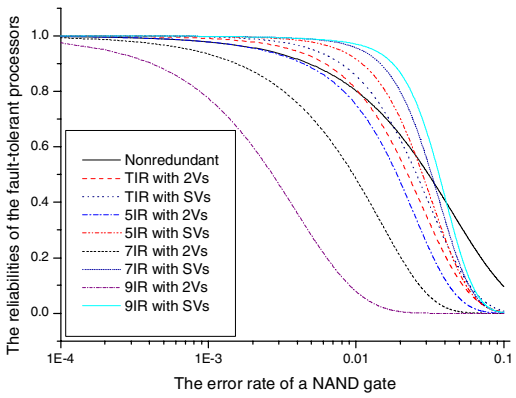

Fig. 12. The reliabilities obtained by simulations of the nonredundant and NIR circuits with 2-level voters (2Vs), following the design in Figure 6, and the single gate voters (SVs), following the design in Figure 7, for R=3 (i.e. TIR), 5, 7 and 9.

The design and implementation of restorative devices (voters) are important for the NIR (TIR) structure. With the use of conventional 2-level voters, as shown in our study, a higher order of NIR presents a worse system reliability, because of a significantly increased size of the voters. With the use of single majority voters, which are favorable for implementations in nanotechnology, the NIR structure offers an advantage over the nonredundant form when the device error rate is not larger than a threshold value. With the single gate voter design, a better system reliability is obtained by using a higher order of NIR.

An NIR implementation of a PE structure, as the one studied in our paper, can be used as the building blocks of a massively parallel, yet fault-tolerant computer architecture. A hierarchical reconfigurability can further be incorporated into the architecture for manufacturing defect-tolerance [23]. Since the basic unit is a fault-tolerant circuit module, the reconfigurability can be limited to the module or higher levels, instead of gate or device level. This would greatly lower the difficulty in system testing and fault diagnosis, which usually present challenges in a large-scale integrated system. Since the NIR, similar as NMR, simultaneously provides the architecture protection from transient errors in system operation, an NIR based parallel architecture may be robust against both manufacturing defects and transient faults for future nanoelectronic systems.

REFERENCES

[1] T. Fountain, *Processor Arrays: Architecture and Applications*, 1987. Academic Press, London, UK.

[2] P. P. Jonker, *Morphological Image Processing: Architecture and VLSI Design,* 1992. Kluwer Publishers, Dordrecht, The Netherlands.

[3] M. J. B. Duff, "CLIP4," *Special Computer Architectures for Pattern Processing*, CRC Press, Boca Raton, Florida, USA, pp.65-86, 1982.

[4] K. E. Batcher, "Design of a massively parallel processor," *IEEE Trans. Comput.*, vol. 29, pp. 836-840, 1980.

[5] S. Kyo, T. Koga and S. Okazaki, "IMAP-CE: a 51.2 GOPS video rate image processor with 128 VLIW processing elements," in *Proc. Int. Conf. Image Processing*, pp. 294-297, 2001.

[6] R. Kleihorst, A. Abbo, A. van der Avoird, M. O. de Beeck, L. Sevat, P. Wielafe, R. van Veen, and H. van Herten, "Xetal: a low-power high-performance smart camera processor," in *proc. IEEE ISCAS 2001*, pp. 215-218, 2001.

[7] *International Technology Roadmap for Semiconductors*, 2003 ed. http://public.itrs.net/.

[8] R. Compañó (ed.), *Technology Roadmap for Nanoelectronics*, 2nd ed. European Commission IST programme Future and Emerging Technologies, 2000.

[9] A. Bachtold, P. Hadley, T. Nakanishi and C. Dekker, "Logic circuits with carbon nanotube transistors," *Science,* vol. 294, pp. 1317-1320, 2001.

[10] Y. Huang, X. Duan, Y. Cui, L. J. Lauhon, K. Kim and C. M. Lieber, "Logic gates and computation from assembled nanowire building blocks," *Science,* vol. 294, pp. 1313-1317, 2001.

[11] C. P. Collier, E. W. Wong, M. Belohradsky, F. M. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams and J. R. Heath, "Electronically configurable molecular-based logic gates," *Science,* vol. 285, pp. 391-394, 1999.

[12] T. J. Fountain, M. J. B. Duff, D. G. Crawley, C. D. Tomlinson and C. D. Moffat, "The use of nanoelectronic devices in highly parallel computing systems," *IEEE Trans. VLSI*, vol. 6, pp. 31-38, 1998.

[13] M. R. Stan, P. D. Franzon, S. C. Goldstein, J. C. Lach and M. M. Ziegler, "Molecular electronics: from devices and interconnect to circuits and architecture," in *Proc. IEEE*, vol. 91, pp. 1940-1957, 2003.

[14] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Studies*, Shannon C.E. & McCarthy J., eds., Princeton University Press, Princeton N.J. pp. 43-98, 1956.

[15] D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems: Design and Evaluation,* 1998. A K Peters, Natick, Massachusetts, USA.

[16] J. R. Heath, P. J. Kuekes, G. S. Snider and R. S. Williams, "A defect-tolerant computer architecture: opportunities for nanotechnology," *Science,* vol. 280, pp. 1716-1721, 1998.

[17] S. C. Goldstein and M. Budiu, "Nanofabrics: spatial computing using molecular nanoelectronics," in *Proc. 28th Int. Symp. Computer Architecture*, pp. 178-189, 2001.

[18] A. DeHon, "Array-based architecture for FET-based nanoscale electronics," *IEEE Trans. Nano*, vol. 2, pp. 23-32, 2003.

[19] S. Spagocci and T. Fountain, "Fault rates in nanochip devices," in *Electrochemical Society Proc.*, vol. 98-19, pp. 582-593, 1999.

[20] K. Nikolić, A. Sadek, M. Forshaw, "Fault-tolerant techniques for nanocomputers," *Nanotechnology, 13*, pp. 357-362, 2002.

[21] J. Han and P. Jonker, "A system architecture solution for unreliable nanoelectronic devices," *IEEE Trans. Nanotechnology*, vol. 1, no. 4, pp. 201-208, 2002.

[22] A. S. Sadek, K. Nikolić and M. Forshaw, "Parallel information and computation with restitution for noise-tolerant nanoscale logic networks," *Nanotechnology, 15*, pp. 192-210, 2004.

[23] J. Han and P. Jonker, "A defect- and fault-tolerant architecture for nanocomputers," *Nanotechnology*, vol. 14, no. 2, pp. 224-230, 2003.

[24] W. H. Pierce, *Failure-Tolerant Computer Design,* Academic Press, 1965.

[25] C. S. Lent, P. D. Tougaw, W. Porod and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, pp. 49-57, 1993.

[26] T. Oya, T. Asai, T. Fukui and Y. Amemiya, "A majority-logic device using an irreversible single-electron box," *IEEE Trans. Nanotechnology*, vol. 2, no. 1, pp. 15-22, 2003.

[27] J. Han and P. Jonker, "Novel computing architecture on arrays of Josephson persistent current bits," in *Proc. 5th Int. Conf. MSM*, pp. 636-639, 2002.

[28] P.P. Jonker, "Why linear arrays are better image processors," in *Proc. 12th IAPR, Int. Conf. on Pattern Recognition*, Volume III, Conf. D: Parallel Computing (ICPR12, Jerusalem, Israel, Oct. 9-13), IEEE Computer Society Press, Los Alamitos, CA, pp. 334-338, 1994.

[29] C. E. Stroud, "Reliability of majority voting based VLSI fault-tolerant circuits," *IEEE Trans. VLSI*, vol. 2, pp. 516-521, 1994.

[30] A. E. Barbour and A. S. Wojcik, "A general, constructive approach to fault-tolerant design using redundancy," *IEEE Trans. Computers*, vol. 38, pp. 15-29, 1989.