A NOVEL MARKOV MODEL FOR THE RELIABILITY PREDICTION OF FAULT TOLERANT NON-HOMOGENOUS MULTIPIPELINES

Hussain Al-Asaad
Computer Engineering Research Laboratory
Department of Electrical & Computer Engineering
University of California, Davis, CA 95616
E-mail: halasaad@ece.ucdavis.edu

ABSTRACT

A novel Markov model for the reliability prediction of fault-tolerant non-homogenous VLSI and WSI multipipeline arrays is presented. The PEs of the array are assumed to fail independently (with a constant failure rate) at different moments and the transition rate between two different error states is constant. A total system failure is reached when the number of working pipelines becomes less than a predetermined number S_m. Thus the reliability of the multipipeline array is defined as the probability of having S(t) greater than or equal to S_m, where S(t) is the number of survived pipelines at time t, and S_m is the minimum number of survived pipelines that is needed for the multipipeline to be considered in a working condition. In addition to predicting the reliability, the Markov model can be used in design optimization to determine the best possible design among multiple alternatives. Several experiments are conducted that demonstrate the ability of the proposed Markov model to predict the reliability and to evaluate various design alternatives.

1 INTRODUCTION

A multipipeline array is a set of identical pipelines each of which consists of several stages of processing elements (PEs) that are separated from each other by interconnection networks. While an individual pipeline is obviously a linear array, the entire architecture can be seen as a rectangular array with a simplified interconnection structure. A general model of multipipelines is shown in Figure 1. Multipipelines are often classified into two categories: Homogeneous and non-homogenous. PEs in homogenous multipipelines perform the same operation and hence they are perfectly identical. Although homogenous multipipelines are rarely used, homogeneity can often be achieved at the expense of extra hardware. In non-homogeneous multipipelines, the different stages of a single pipeline perform different operations and the PEs are therefore different. So, in non-homogenous multipipelines the homogeneity is found *column-wise*. Hereinafter, the term "multipipelines" refers to non-homogeneous multipipelines.

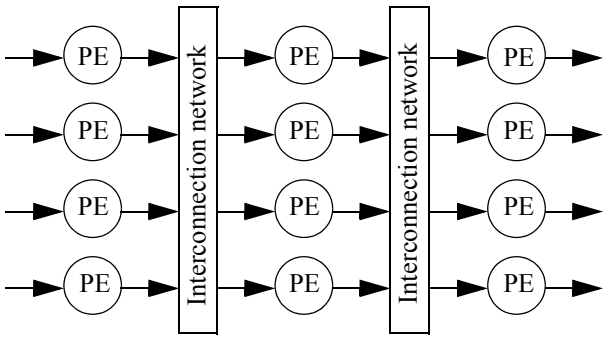

Figure 1 A general model of a multipipeline.

The simplest form of the multipipeline is a one with feed-through connections and hence has no fault-tolerant capabilities. The most complex form of the multipipeline uses a crossbar interconnection network between the stages. As the connectivity of the interconnection network increases, the hardware required to implement the network increases which in turn increase the probability of failure in the interconnection network. Hence, a balanced trade-off between the degree of fault tolerance and the complexity of the interconnection network need to be located.

Multipipelines are often used to perform parallel pipelined operations with efficient performance. From general purpose vector supercomputers to application specific digital signal processing and cellular arithmetic arrays, multipipelines are currently being used. For example, they are currently being used in bit

serial digital signal processing arrays/transforms [1].

Very large scale integration (VLSI) and wafer scale integration (WSI) technologies are most advantageous when used to implement regularly structured systems such as large arrays of identical processing elements. As integration level increases and the sizes of arrays grow larger, the possibility of a single fault or multiple faults occurring in a VLSI or WSI array increases. These faults can occur during the manufacturing process as well as during the operational lifetime of an array. If an array is not fault tolerant, the failure of a single element can cause the entire array to fail. On the other hand, the array might be able to operate in a fault-tolerant reconfigurable structure, where it is designed to tolerate some of the faults. For example, in the presence of faults, the multipipeline may be designed to recover k needed pipelines out of N supplied ones. So, if a vector processor uses at least six pipelines and eight of them are supplied, then a fatal failure is reached when three out of the eight pipelines are faulty. This restructuring of the array can be performed at fabrication time to enhance yield or during normal operation to improve reliability.

The problems of reconfiguring functional pipelines out of an array with faults have received much attention [1][2][3][4][5][6][7][8][9]. Some reconfiguration algorithms assume fault-free switches and interconnections. A multi-phase representative algorithm is described in [2]. Each phase of the algorithm consists of sequentially setting rows of switches. For an $N \times M$ multipipeline array, N sequential phases are required. Other algorithms consider switch and interconnect faults. The algorithm proposed in [8] does not produce optimal results and is complex and difficult to be implemented distributively.

Another problem with known reconfiguration algorithms is that the interconnection lengths of a reconfigured array could become significantly longer than the original array. This can decrease the performance benefits of implementing the pipelines on a single VLSI or WSI chip, since multipipelines are a synchronous design where the clock is set to accommodate the longest delay of interstage connections. Furthermore, since it is not possible to know a priori the length of interstage connections in the pipelines, then all interconnections need to be supported with powerful buffers capable of driving the worst case interstage paths. This can impose very significant area, power, and delay penalties on the multipipeline design. Due to the above, several reconfiguration techniques ensure that interstage connections are probabilistically bounded [1][2].

A major weakness of known reconfiguration algorithms is that they are not simple enough to be implemented with little hardware and to be executed in a very short time, especially when the reconfiguration is performed on-line during normal operation. For example, the reconfiguration algorithm presented in [3] achieves optimal solution based on finding the maximum flow in a flow network but suffers from its complexity and difficulty to be implemented in a distributed fashion.

As discussed above, previous designs of multipipelines are characterized by variable interstage connection length dependent on the fault distribution, complex switching element that forbids the assumption of fault-free switches, and multi-phase sequential reconfiguration algorithms. A new design is presented in [11] that guarantees a constant and fault-distribution independent interstage length. The design is characterized by its simplicity—the switching element is replaced by a simple two-input multiplexer—and has a parallel distributed reconfiguration algorithm.

In general, the following issues need to be considered in designing efficient fault-tolerant multipipelines:

- *Architecture*: The interconnection network between the columns of the processing array should support fault-tolerant capabilities. It should be simple enough so it does not add penalties on the array performance. Also, the interstage path length should be minimized.
- *Diagnosis*: The diagnosis algorithm that detect defects/faults in both the network and the processing elements should be simple so that the testing hardware is kept at a minimal.
- *Reconfiguration*: The reconfiguration algorithm should give good harvest rate and should be simple enough so that it can be easily implemented and executed in a short time.

The reliability of a multipipeline design is calculated using Markov models. The PEs are assumed to fail independently with a constant failure rate λ measured in failures per PE per unit time. A system failure is reached when the number of working pipelines becomes less than a certain number S_m. Thus the reliability is defined as $R(t) = Prob\{S(t) \geq S_m\}$, where

$S(t)$ is the number of survived pipelines at time t, and S_m is the minimum number of survived pipelines that is needed for the multipipeline to be considered in a non-fatal failure condition.

The rest of the paper is organized as follows. In Section 2, we describe the details of the new Markov model for multipipelines. In Section 3, we present an evaluation of the reliability of several existing designs using the new Markov model.

2 NEW MARKOV MODEL

The Markov model for reliability prediction requires two assumptions [1][15]: PEs fail independently in different moments, and the transition rate between two different error states of a system of PEs (multipipeline) is constant.

To develop the model of the multipipeline, consider first the simple case of a 4×3 multipipeline. The Markov model for the multipipeline with $S_m = 2$ is shown in Figure 2. Each circle in that figure represents an ensemble of states that is represented by (α, β), where α represents the number of survived pipelines and β represents the number of faulty PEs. Initially, the multipipeline is in ensemble (4,0) and finally, the multipipeline is in an ensemble $(1, \gamma) \equiv F$, where $\gamma > 2$ and F is the fatal failure ensemble. There are many states of the multipipeline that have 3 survived pipelines with 1 faulty PE. In fact, the ensemble (3,1) has 4×3 states. On the other hand the ensemble (4,0) has only one state.

The first fault in the multipipeline leads to the loss of one pipeline as shown in Figure 2. The second fault in the multipipeline could lead to losing another pipeline if the first and second faults occurred in PEs

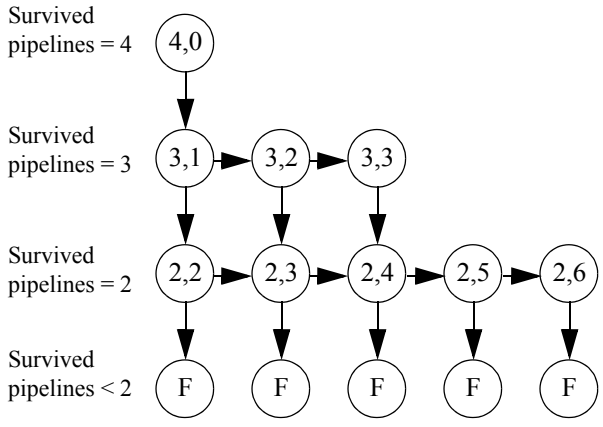

Figure 2 Markov model for a 4x3 multipipeline.

of the same column. Otherwise, no additional pipeline is lost. In general, after the first fault, a second fault may or may not lead to a pipeline loss.

Consider the most general case of $N \times M$ multipipeline where a fatal failure is reached when the number of survived pipelines is less than S_m. By inspection, the ensembles in the Markov model are as follows:

one fault-free ensemble which has a single state

$1 \times (M-1) + 1$ ensembles each has $N-1$ fault-free pipelines

$2 \times (M-1) + 1$ ensembles each has $N-2$ fault-free pipelines

...

$k \times (M-1) + 1$ ensembles each has $N-k$ fault-free pipelines

....

$(N-S_m) \times (M-1) + 1$ ensembles each has S_m fault-free pipelines

$(N-S_m) \times (M-1) + 1$ fatal failure ensembles

Hence, the total number of ensembles of the Markov model is:

$$\begin{aligned} S_T &= 1 + 1 \times (M-1) + 1 + 2 \times (M-1) + \\ &\quad 1 + \ldots + (N-S_m) \times (M-1) + 1 \\ &= (M-1) \times \{1 + 2 + \ldots + (N-S_m)\} + \\ &\quad (N-S_m) + 1 + (N-S_m) \times (M-1) + 1 \\ &= (M-1) \times \frac{\{(N-S_m)+1\}}{2} \times (N-S_m) + \\ &\quad (N-S_m) + (N-S_m) \times (M-1) + 2 \end{aligned}$$

$$S_T = (N-S_m) \times \left[(M-1) \times \frac{\{(N-S_m)+1\}}{2} + M \right] + 2$$

If $M = N$, then S_T will be $O(N^3)$. This shows the rapid growth of complexity of the Markovian modeling. Initially in the Markov model, the multipipeline is in ensemble $(N, 0)$ and finally, the multipipeline is in an ensemble $(S_m - 1, \gamma) \equiv F$, where $\gamma > (N - S_m)$. In general, each ensemble of the Markov model can be represented as shown in Figure 3.

From each ensemble (p, f) of the Markov model, there exists q transitions (PE failures) to other ensembles of which r lead to a pipeline loss. Since the probabilities of failure of the PEs are constant, equal, and independent, then the rate of pipeline failures from ensemble (p, f) is the number of live PEs $(N \times M - f)$ times the PE failure rate (λ) times the transitional fraction r/q (F_v). The transitional fractions F_v', F_h',

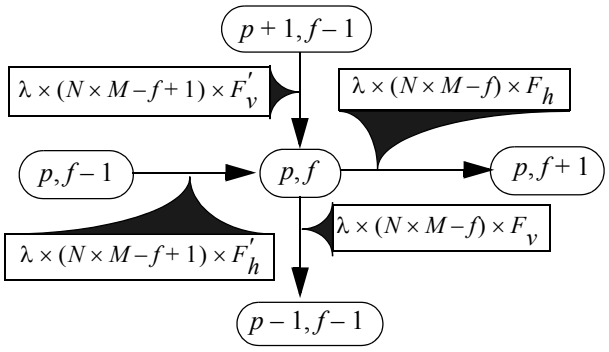

Figure 3 General Markov model for an ensemble.

F_v, and F_h are important parameters that become increasingly burdensome to obtain analytically as N increases. For small number of PEs, enumerating the states and grouping them into ensembles leads to finding the actual values of the transitional fractions. For medium and large number of PEs, the enumeration technique is not practical. Hence, simulation is used to determine the transitional fractions. The transitional fraction F_v' is the fraction of PE failures which lead to a pipeline loss while the multipipeline is in the ensemble $(p + 1, f - 1)$. Similarly, F_v is the fraction of PE failures which lead to loss of a pipeline in the ensemble (p, f). On the other hand, F_h' is the fraction of PE failures which do not lead to a pipeline loss in the ensemble $(p, f - 1)$. Similarly, F_h is the fraction of PE failures which do not lead to a pipeline loss in the ensemble (p, f). Let $P_{(a,b)}(t)$ be the probability of being in ensemble (a, b). Then the following equation relates the ensembles in Figure 3.

$$\frac{dP_{(p,f)}(t)}{dt} = -\lambda \times (N \times M - f) \times P_{(p,f)}(t) +$$

$$\lambda \times (N \times M - f + 1) \times [F_v' \times P_{(p+1,f-1)}(t) + F_h' \times P_{(p,f-1)}(t)]$$

Ensemble (p, f) should be provided by F_h' and F_v' in order to determine the probability of being in it at time $t + \Delta t$. Note that this probability is independent of F_v and F_h because $F_v + F_h = 1$.

To get the transitional fractions F_v' and F_h' for each ensemble, a simulation procedure is used. For each ensemble, two variables N_v and N_h are defined. In ensemble (p, f), $N_v(p, f)$ is the number of times the ensemble is visited from ensemble $(p + 1, f - 1)$ and $N_h(p, f)$ is the number of times the ensemble is visited from ensemble $(p, f - 1)$. Each time the ensemble is visited either $N_v(p, f)$ or $N_h(p, f)$ is incremented. The simulation procedure of getting F_h' and F_v' is thus as shown in Figure 4.

Repeat 1000 times
    Initialize the multipipeline to fault-free state
    Repeat
        Generate a fault
        Move to the corresponding ensemble
        Increment either Nv or Nh
    Until fatal-failure
Until done
for all ensembles do **begin**
    Fv' = Nv(p,f) / [(Nv(p+1,f-1)+Nh(p+1,f-1)]
    Fh' = Nh(p,f) / [(Nv(p,f-1)+Nh(p,f-1)]
end

Figure 4 A simulation algorithm to determine the transitional fractions.

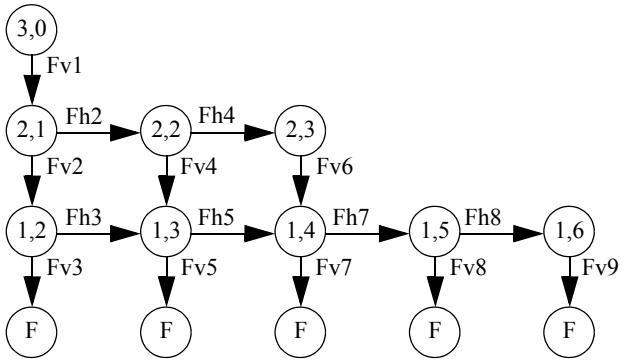

Figure 5 The transitional fractions in the Markov model for a 3x3 multipipeline.

To verify that the values of F_v' and F_h' determined using the above procedure converge to their respective exact values, the transitional fractions are derived manually for the case of 3×3 multipipeline with $S_m = 1$. The Markov model for the multipipeline is shown in Figure 5. Since the sum of the transitional fractions going out from an ensemble sums to 1, comparing the vertical transitional fractions calculated to that determined by simulation is sufficient. Table 1 shows the exact values of the transitional fractions as well as the simulation values with their error percentage. The simulation is done using 1000 and 5000 iterations. It is clear that as the number of iterations increase, the simulation values tend to their corresponding exact values. From Table 1, we can see that the maximum percentage error is 1.06% with 5000 iterations. Hence, using simulation to get the transitional fractions is sufficient for determining the reliability of the multipipelines.

Table 1 A comparison of the transitional fractions determined by simulation to their corresponding exact values.

Transitional fractions	Exact values	Using 1000 iterations		Using 5000 iterations	
		Simulation values	Percentage error	Simulation values	Percentage error
FV1	1.0000	1.0000	0.0000	1.0000	0.0000
FV2	0.2500	0.2492	0.3200	0.2506	-0.2240
FV3	0.1429	0.1388	2.8400	0.1435	-0.4780
FV4	0.5714	0.5692	0.3900	0.5721	-0.1140
FV5	0.2407	0.2323	3.5062	0.2382	1.0595
Fv6	1.0000	1.0000	0.0000	1.0000	0.0000
FV7	0.4439	0.4383	1.2621	0.4404	0.8003
FV8	0.6842	0.6783	0.8638	0.6784	0.8492
Fv9	1.0000	1.0000	0.0000	1.0000	0.0000

3 Simulation Results

We compute the reliability of the following multipipeline designs:

- NEW: The most recent multipipeline design presented in [11].
- OLD: The classical multipipeline design used by Gupta et al. in [2].
- MIN: A straight through non-fault tolerant design.
- MAX: A design with crossbar interconnections between stages.

To evaluate the reliability of these designs using the Markov model, simulation experiments are performed over an 8×8 multipipeline. The reliability is defined to be the probability of being in a non-fatal failure state at time t. The results of this simulation are shown in Figure 6, Figure 7 and Figure 8. In Figure 6, the reliability of the NEW multipipeline is plotted for different values of S_m. It is clear (trivial) that as S_m decreases, the reliability increases. From Figure 7 and Figure 8, we can see that NEW design has a good reliability compared to OLD design especially when S_m is large (i.e., when we have small amount of hardware redundancy). Also, we can see that both NEW and OLD are far better than the straight-through multipipeline MIN.

Another way for comparing the reliability is by comparing the mean time to failure (MTTF). The results of simulation on the 8×8 multipipeline are shown in Figure 9. In this figure, the MTTF of the

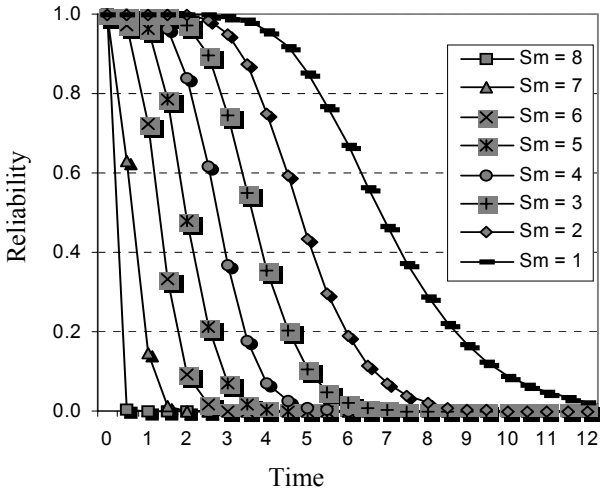

Figure 6 The reliability of an 8x8 NEW multipipeline with a 0.1 PE failure rate.

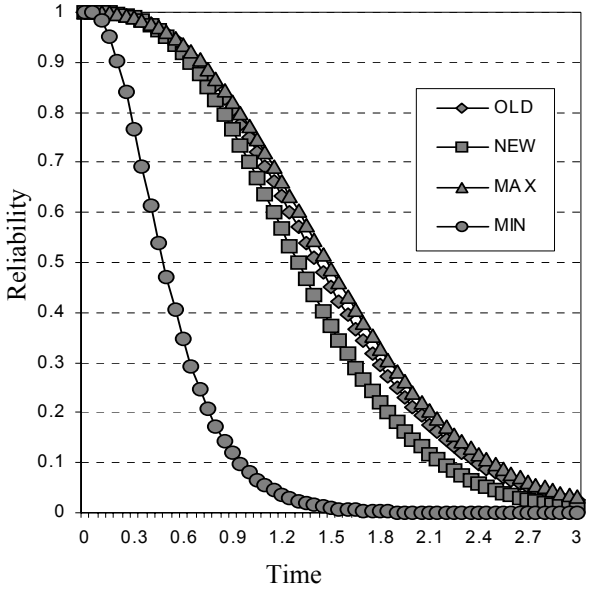

Figure 7 The reliability of an 8x8 multipipeline with a PE failure rate of 0.1 and S_m = 6.

NEW design approaches that of OLD design for large values of S_m. In fact, when S_m = 7, the MTTF for NEW, OLD, and MAX are equal.

In summary, a novel Markov model for the reliability prediction of general fault-tolerant multipipelines is derived and a simulation procedure to determine the transitional fractions using this model is presented. The model can be used to determine the best possible design among multiple alternatives. Several experiments are conducted that demonstrate

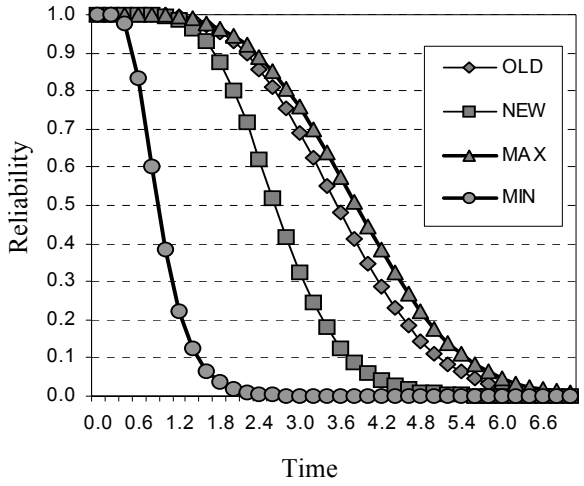

Figure 8 The reliability of an 8x8 multipipeline with a PE failure rate of 0.1 and S_m = 4.

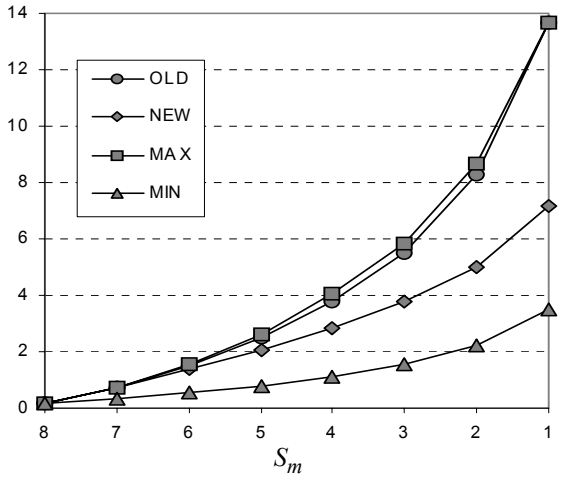

Figure 9 Mean time to failure of an 8x8 multipipeline with a 0.1 PE failure rate.

the ability of the new Markov model to predict the reliability and to evaluate various design alternatives.

ACKNOWLEDGMENTS

The author wishes to thank Prof. James Feldman and Mankuan Vai for their initial comments on this research.

REFERENCES

[1] R. Negrini, M. Sami, and R. Stefanelli, *Fault Tolerance Through Reconfiguration in VLSI and WSI Arrays*, The MIT press, 1989.

[2] R. Gupta, A. Zorat, and I. Ramakrishnan, "Reconfigurable multipipelines for vector supercomputers", *IEEE Transactions on Computers*, Vol. 38, pp. 1297-1307, September 1989.

[3] H. Lin, F. Lombardi, and M. Lu, "On the optimal reconfiguration of multipipeline arrays in the presence of faulty processing and switching elements", *IEEE Transactions on VLSI Systems*, Vol. 1, No. 1, pp. 76-79, March 1993.

[4] P. Koo, F. Lombardi, and Y. Shen, "Approach for the reconfiguration of multipipeline arrays", *IEE Proceedings-E*, Vol. 138, No. 3, pp. 131-137, May 1991.

[5] Y. Choi, "Reconfigurable VLSI/WSI multipipelines", *Parallel Computing,* Vol. 17, pp. 941-952, 1991.

[6] P. Kogge, *The Architecture of Pipelined Computers*, New York, McGraw-Hill 1981.

[7] Y. Choi, "Fault diagnosis of reconfigurable multipipelines using boundary scans", *Computers and Electrical Engineering*, Vol. 18, No. 2, pp. 119-130, 1992.

[8] Y. Choi, "Reconfigurable multipipelines", *Proc. International Conference on Parallel Processing*, 1991, pp. 556-570.

[9] M. Chean and J. Fortes, "A taxonomy of reconfiguration techniques for fault-tolerant processor arrays", *IEEE computer*, pp 55-69, January 1990.

[10] W. Shi, M.-F. Chang, and W. K. Fuchs, "Harvest rate of reconfigurable pipelines", *IEEE Transactions on Computers*, Vol. 45, pp. 1200-1203, October 1996.

[11] H. Al-Asaad, M. Vai, and J. Feldman "Distributed reconfiguration of fault tolerant VLSI multipipeline arrays with constant interstage path lengths", *Proc. of International Conference on Computer Design*, 1994, pp. 75-78.

[12] H. Jagadish, R. Mathews, T. Kailath, and J. Newkirk, "A study of pipelining in computing arrays", *IEEE Transactions on Computers*, Vol. C-35, pp. 431-439, May 1986.

[13] H. Jordan, "Experience with pipelined multiple instruction streams", *Proc. of the IEEE,* Vol. 72, pp. 113-123, January 1984.

[14] G. Sohi, M. Franklin, and K. Saluja, "A study of time-redundant fault tolerance techniques for high-performance pipelined computers", *Proc. Fault Tolerant Computing Symposium*, 1989, pp. 436-443.

[15] B. Johnson, *Design and Analysis of Fault Tolerant Digital Systems*, Addison-Wesley Publishing Company, 1989.

[16] M. Tchuente, *Parallel Computation on Regular Arrays*, Manchester University Press, 1991.