Proceedings of the Twentieth Annual Hawaii International Conference on System Sciences, 1987.

WAFER-SCALE INTEGRATION OF MULTI-PROCESSOR SYSTEMS*

I. Koren
Dept. of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003
on leave from the Technion, Haifa, Israel

Z. Koren
Dept. of Computer Science
Technion - Israel Institute of Technology
Haifa 32000, Israel

D.K. Pradhan
Dept. of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA 01003

Abstract

Recent advances in the integrated circuit technology and development of advanced computer-aided design tools (like silicon compilers) enable the design and implementation of complex systems on a single large area silicon chip or even an entire wafer.

The design of complete systems on one piece of silicon can benefit from the main advantages of wafer-scale integration like faster communications, less packaging steps and a less severe pin limitation. However, this "wafer-scale integration" introduces new problems which do not exist when the different elements of a system are implemented on separate silicon chips. First, defective system elements can be neither repaired nor discarded and they must be dealt with by introducing fault-tolerance into the architecture of the designed system. Secondly, all designed elements must share the same silicon area and the same allowed power dissipation. Hence, we have to determine how to partition the available area and the total allowed power dissipation among the different types of elements in order to optimize some objective function. These design problems may lead to changes in well-established system architectures and development of new ones which may be more appropriate to wafer-scale integration.

In this paper we illustrate this new design environment through the example of closely-coupled multi-processor systems and introduce several objective functions one may wish to optimize in this environment.

1. INTRODUCTION

Recent advances in VLSI technology and development of new computer-aided design tools like silicon compilers enable the design and implementation of complex computer systems on a single silicon chip. This may be a large-area chip (compared to chip areas nowadays) or even an entire wafer. The design of complete systems on one piece of silicon introduces new design problems which do not exist when the different elements of the system are implemented on separate silicon chips. As an example for this new design environment we consider here closely-coupled multi-processor computer systems.

A multi-processor system implemented on a single silicon chip can benefit from the main advantages of wafer-scale integration, e.g, faster communications among the various system elements and less expensive manufacturing steps (i.e., there is no need to dice the wafer into individual chips and bond their pads to external pins). In addition, by implementing the entire system on one piece of silicon we turn all previously external interconnections into internal ones avoiding this way the problem of pin limitation (e.g., [FrWa82]).

However, the well-established architectures of closely-coupled multi-processor systems are not always suitable for direct VLSI implementation. This is due to the fact that different constraints have to be dealt with in this new design environment. First, there is a finite amount of silicon area which all types of system elements must share; a constraint that does not exist when every system element is manufactured on a separate silicon chip. This implies that we have to determine how many elements of each type (e.g., processors, memory modules, etc) we should include in our design, given the area required by each of these elements. This should be determined in such a way so that some system objective function (e.g., some performance measure) will be optimized.

Secondly, the fact that all the elements of the multi-processor system share the same piece of silicon poses a restriction on the total amount of power that can be dissipated by the system. This restriction is in most cases more severe than the equivalent one for ordinary multi-processors. If not all system elements dissipate the same amount of power, the above restriction may result in the inclusion of less than the optimal number of the high-power elements.

Thirdly, in the traditional assembling of multi-processors only defect-free ICs are used while the defective ones are

*This work was supported in part by NSF under contract DCR-85-09423.

discarded. This can not be done in the environment of a wafer where several system elements are bound to be defective and they can neither be repaired nor discarded. This has two important implications: redundancy must be incorporated into the design so that we will be able to use the system in the presence of defects, and then, the system has to be designed in a way that will allow us to interconnect most (if not all) of the defect-free elements to form an operational multi-processor. Thus, we must devise fault-tolerant architectures for these multi-processor systems to be implemented on one piece of silicon.

These three factors, the finite amount of silicon area, the limited power dissipation and the need for fault-tolerance, may lead to different optimal designs of existing architectures and development of new architectures.

2. REDUNDANCY IN MULTI-PROCESSORS

The closely-coupled multi-processor systems considered here consist of several processors, memory modules and a network interconnecting them. System level redundancy can be easily applied to the processors and memory modules in a multi-processor system by adding such redundant elements. The problem of introducing redundancy into the network interconnecting the processors and memories (including the redundant ones) is more involved. Here, fault-tolerance must be introduced into the architecture of the network to allow reconfiguration upon a fault or defect in a switching element or an internal link so that most (if not all) of the fault-free processors and memories will be interconnected. The best solution to be adopted might clearly depend on the topology of the interconnection network.

Several interconnection networks have been suggested for multi-processor systems, such as the crossbar, the multiple bus and many types of multi-stage networks (some of which were shown to be topologically equivalent).

Crossbar networks have full-interconnection, non-blocking characteristics but were usually considered prohibitively expensive for systems with a large number of processors. Consequently, other interconnection networks have been proposed to provide objective-effective processor-memory communication. These include the multi-stage networks and the multiple bus ones. Both have a lower maximum bandwidth compared to that of a crossbar. However, the potential bandwidth of the crossbar can not anyway be fully utilized due to memory conflicts (i.e., two or more processors attempting to access the same memory module). Therefore, the replacement of a crossbar by a multi-stage or a multiple bus network may only slightly degrade the system performance. For example, in [LaVa82] it has been shown that for a multiple bus network with B busses, P processors and M memories, the degradation in performance with respect to the crossbar is only 5% if $B \cong P/2$ (and $M \cong P$).

On the other hand, the traditional comparison of complexity between the crossbar and its alternatives may not be valid in the wafer-scale integration environment. In the ordinary integrated circuit environment, the complexity of a network is measured by the number of its basic elements which are in our case switches (2 by 2 routers). The number of these elements which are required in a crossbar network is substantially higher than that in a multi-stage network. In the wafer-scale integration environment the total silicon area required for the layout of the interconnection network is a more appropriate measure of network complexity rather than the number of switching elements. Since the connection paths among the switches use substantial amounts of silicon area, the total area taken by a crossbar network is comparable to that taken by a multi-stage network [Fran81].

Consequently, in the wafer-scale integration environment all three kinds of interconnection have to be considered and appropriate redundancy schemes have to be developed, e.g., [AdSi82], [Geor84], [KuRe85], [PaLa83] and [TzYe85].

Multi-stage networks are very sensitive to failures of any kind. They provide a unique path between any input module and any output module and therefore, a single fault in any internal switch or link will render some outputs unreachable from certain inputs. Consequently, schemes for introducing fault-tolerance into the architecture of these interconnection networks have been suggested in several recent publications.

If the main purpose of the added redundancy is yield enhancement we may consider the possibility of increasing the feature size of the silicon layout to reduce the probability that manufacturing defects will cause circuit failures. Defects that are too small compared to the width of lines or the spacing between lines, may be harmless to the electrical performance of the circuit. Therefore, increasing the feature size will reduce the percentage of defects that cause circuit failures. However, increasing the feature size of all the devices within a system element will result in a larger total area of that element and as was shown in [Stap84] the increased area may cancel the effect of reduced failure density. Still, increasing feature size may have the desired effect of yield enhancement in restricted situations. For example, when laying out an interconnection bus of a given length, a larger feature size will result in an increased width of the bus but may leave the length unchanged. Thus, the change in total area will not cancel out the effect of reduced failure density.

Let w and s denote the width of a single conductor and the spacing between two adjacent conductors within a bus, respectively. Let β be the factor by which we increase the feature size. Thus, both w and s are increased by β and if the bus length remains unchanged the total area of the bus is increased only by β. It has been shown in [Stap84] that the probability that a photolithographic defect will result in either an open circuit in a conducting line or a short circuit between two adjacent lines is given by,

$$\Theta = \frac{\chi_0^2}{2w(2w + s)} \qquad (2.1)$$

where χ_0 is the defect size with the highest relative defect density.

From the above expression it is apparent that the probability Θ is reduced by β^2 when the feature size is increased by β and as a result the total number of expected circuit failures within the bus element (of increased area) is reduced by β.

To evaluate the effectiveness of all the above redundancy schemes (and different ones) for a given system architecture, and to determine the optimal amount of redundancy for each scheme we need an objective function. Examples of such objective functions are presented in the next section.

3. OBJECTIVE FUNCTIONS FOR MULTI-PROCESSORS ON A WAFER

Objective functions for multi-processors on a wafer allow us to compare alternative architectures and to determine for each architecture the optimal values of the parameters like number of processors, number and size of memory modules, topology of the interconnection network, etc.

The objective function to be used may be of two different types. The first type includes those objective functions concerned only with the performance of the system after the manufacturing of the wafer has been completed. The objective functions of the second type are concerned also with the performance of the system throughout its mission time. For the first type of objective functions we have to consider only manufacturing flaws while for the second one, operational faults need also be considered.

Let $C(n_1, n_2, ..., n_k)$ be a measure of the performance of a given architecture of a multi-processor system, where k is the number of different types of elements and n_i is the number of fault-free elements of type i. The quantity n_i satisfies, $0 \le n_i \le N_i$, $i = 1, 2, ..., k$; where N_i is the total number of elements of type i which were originally designed.

Let $P(n_1, n_2, ..., n_k, t)$ be the probability that at time t the number of operational elements of type i is n_i, $i = 1, 2, ..., k$. $P(n_1, n_2, ..., n_k, 0)$ is the probability that after manufacturing $(t = 0)$, the number of defect-free elements of type i is $n_i, i = 1, 2, ..., k$. This probability depends on the distribution of defects in the wafer, while $P(n_1, n_2, ..., n_k, t)$ for $t > 0$ depends also upon the distribution of operational faults in the different system elements throughout the mission time. Since defects and faults in separate parts of the silicon wafer are independent, we have

$$P(n_1, n_2, ..., n_k, t) = \prod_{i=1}^{k} P_i(n_i, t) \tag{3.1}$$

where $P_i(n_i, t) = Pr\{\ n_i$ out of N_i elements of type i are fault-free at time t $\}$. An expression for $P_i(n_i, t)$ will be derived later on.

Based on the $P_i(n_i, t)$'s and some performance measure $C(n_1, n_2, ..., n_k)$, we can define the following objective function $F(N_1, N_2, ..., N_k, t)$ to be optimized,

$$F(N_1, N_2, ..., N_k, t)$$

$$= C(n_1, n_2, ..., n_k) \sum_{n_1 = l_1}^{N_1} \sum_{n_2 = l_2}^{N_2} \cdots \sum_{n_k = l_k}^{N_k} \cdot \prod_{i=1}^{k} P_i(n_i, t) \tag{3.2}$$

where only ICs with at least l_i fault-free elements of type i are accepted.

This objective function is the expected value of the performance measure at time t, and we have to find the values of $N_1, N_2, ..., N_k$ that optimize it under the constraints,

$$\sum_{i=1}^{k} N_i A_i \le A \tag{3.3}$$

$$\text{and} \sum_{i=1}^{k} N_i E_i \le E \tag{3.4}$$

where A is the total chip area, A_i is the silicon area of the type i element, E is the maximum allowed power dissipation and E_i is the power dissipated by a type i element.

If $C(n_1, n_2, ..., n_k) = 1$ then the above objective function $F(N_1, N_2, ..., N_k, 0)$ (as defined in (3.2) for $t = 0$), becomes the yield. For $t > 0$ and $C(n_1, n_2, ..., n_k) = 1$, the function $F(N_1, N_2, ..., N_k, t)$ becomes the reliability.

The objective function in (3.2) measures the expected performance of the system at time instant t. If the average performance of the system during the time interval $[0, t]$ is of interest, we may integrate $F(N_1, N_2, \ldots, N_k, t)$ to form the objective function,

$$\frac{1}{t} \int_0^t F(N_1, N_2, ..., N_k, s)\, ds$$

All the above objective functions are appropriate when the total chip area A is predetermined and we only have to decide what part of this area should be devoted to each type of elements. A different situation may arise when the chip area can be changed so as to optimize some wafer-level objective function.

The expected performance of a wafer can be defined as the expected performance of a chip times the number of chips per wafer. Let $A_{min} = \sum_{i=1}^{k} l_i A_i$ be the minimum area of a chip. If we increase the chip area to A by adding some redundancy, then the area increase factor is,

$$\gamma = \frac{A}{A_{min}} = 1 + \frac{\sum_{i=1}^{k}(N_i - l_i)A_i}{A_{min}} \tag{3.5}$$

This factor, which is called the redundancy factor [KoBr84], determines the reduction in the number of chips per wafer when redundancy is added to the chip. Consequently, the function $\frac{1}{\gamma} F(N_1, N_2, \ldots, N_k, t)$ for any $F(N_1, N_2, ..., N_k, t)$ may serve as a wafer-level objective function.

We next derive an expression for $P_i(n_i, t)$. At time t there are $(N_i - n_i)$ faulty elements of type i. These consist of elements that became defective during the manufacturing process and elements that suffered operational faults in the time interval $[0, t]$. Let m_i denote the number of defect-free elements of type i at $t = 0$, then

$P_i(n_i, t) = \sum_{m_i = n_i}^{N_i} P_i(m_i, 0) \cdot Pr\{(m_i - n_i)$ out of m_i elements became faulty in $[0, t]\}$

If the operational faults in type i elements occur at a constant failure rate λ_i, then the last term equals,

$$\left[{m_i \atop n_i} \right] (e^{-\lambda_i t})^{n_i} (1 - e^{-\lambda_i t})^{m_i - n_i}\ p_i^{m_i - n_i} \tag{3.6}$$

where p_i is the probability that the system recovers (reconfigures) successfully given that a fault in type i element has occurred [KoBr84].

One can verify that equation (3.6) can be derived from the general equation in [KoPr85] when substituting the proper expressions for the transition rates.

To derive an expression for $P_i(m_i,0)$ we will make assumptions similar to those made in [KoBr84] and [KoPr85]. First, we assume that the random spot defects (caused by minute particles deposited on the wafer) which constitute the majority of fabrication defects [StAr83] will affect a silicon area which is substantially smaller than the area of a system element like a processor, a memory, or a bus.

Secondly, we assume that the distribution law obeyed by the manufacturing defects is the generalised negative binomial distribution. This is one of the distributions suggested in the literature for fabrication defects like the Poisson distribution, the binomial distribution and others. Under proper assumptions each one of these distributions can be used and the "correct" one is the one that fits the data best. We adopt here the generalised negative binomial distribution since it has been shown to agree with experimental results, mainly because defects are allowed to cluster rather than being evenly distributed throughout the wafer [StAr83]. Still, our subsequent results remain valid if some other distribution is selected.

The negative binomial distribution has two parameters; d is the average defect density and α is the defect clustering parameter. A low value of α can be used to model severe clustering of defects on a wafer, while for $\alpha -> \infty$ we obtain the Poisson distribution. These two parameters depend on the number and complexity of manufacturing steps performed. Consequently, different system elements might have different values for these two parameters. Let d_i and α_i be the two parameters for type i system elements, then the probability of having x_i defects in the system elements of type i on a chip is,

$$Pr\{X_i = x_i\} = \frac{\Gamma(x_i+\alpha_i)}{x_i!\Gamma(\alpha_i)} \cdot \frac{(\frac{N_iA_id_i}{\alpha_i})^{x_i}}{(1+\frac{N_iA_id_i}{\alpha_i})^{\alpha_i+x_i}} \qquad (3.7)$$

where N_iA_i is the chip area devoted to the N_i elements of type i.

The parameters d and α of the yield equation are estimated either by monitoring many wafers or from a few carefully placed test chips on each wafer. Another technique for estimating the parameters of the yield equation has been recently proposed by Seth and Agrawal [SeAg84] based on the commulative coverage of test patterns applied to the chips after manufacturing.

Note that in equation (3.7) we should have used the density of circuit failures rather than the density of manufacturing defects. Not all manufacturing defects cause erroneous behavior of the circuit. For example, a defect in the outer area of the chip which is usually occupied by bonding pads, may be harmless. To consider circuit failures instead of fabrication defects, we have to multiply the defect density d by the probability Θ that a circuit failure is caused by a defect. For convenience, we will in what follows still refer to manufacturing defects (rather than circuit failures) with average density d which equals the original defect density multiplied by the probability Θ.

The required expression for $P_i(m_i,0)$ is the probability that all x_i manufacturing defects in type i elements will be restricted to exactly $(N_i - m_i)$ elements. Following the notation in [KoPr86] we define,

$$Q_{x,j}^{(N)} = Pr\{x \text{ defects are distributed into exactly j out of N elements / There are x defects }\}$$

which equals,

$$Q_{x,j}^{(N)} = \sum_{k=0}^{j} (-1)^k \binom{N}{k, j-k, N-j} [\frac{j-k}{N}]^x$$

$$\text{for } x \geq j \text{ and } 0 < j \leq N \qquad (3.8)$$

where $\binom{N}{k,j-k,N-j}$ is the multinomial coefficient. For $x < j$ we have $Q_{x,j}^{(N)} = 0$ and for $x = j = 0$ $Q_{0,0}^{(N)} = 1$.

Using the probability $Q_{x,j}^{(N)}$ we obtain the following equation for $P_i(m_i,0)$,

$$P_i(m_i,0) = \sum_{x_i=(N_i-m_i)}^{\infty} Q_{x_i,(N_i-m_i)}^{(N_i)} \cdot Pr\{X_i = x_i\} \qquad (3.9)$$

We may substitute the last term in equation (3.9) by equation (3.7) or a similar expression for any other defect distribution.

Similarly to equation (3.6) we may multiply (3.9) by the conditional probability that an element can be bypassed (isolated) given that it is defective. This allows us to consider less than perfect procedures for locating defective elements and reconfiguring them out of the system.

4. EXAMPLE - A MULTIPLE BUS MULTIPROCESSOR

As an example for a multi-processor system on a single large area VLSI chip, consider a multiple bus multiprocessor system. Let P, M, and B denote the number of homogeneous processors, memory modules and interconnecting busses, respectively, that are operational (fault-free). Such a system is illustrated in Figure 1 where each processor is connected to all busses and each bus to all memory modules, so that a processor can access any memory module through any of the busses. Following the notation used in [MaGe82], we refer to this multiprocessor as a $P * M * B$ system.

The behavior of the system is characterized by the following two parameters. Let the time between two consecutive memory references by a single processor be a random variable with an exponential distribution having a mean $1/\delta$. Also, let the memory transfer time be exponentially distributed with mean $1/\mu$.

We wish to find an expression for the performance of a $P * M * B$ system. Such an expression will allow us to determine the optimal number of processors, memory modules and interconnection busses to be designed in the VLSI chip so that the corresponding objective function will be maximised.

We may select as performance measure the *processing power* of the $P * M * B$ system which is defined here as the expected number of active processors, i.e., processors which are executing their task and not idle while waiting to access a common memory module. Other performance measures like the average cycle time and the instruction execution rate can be simply derived from the processing power measure [MaGe82].

To calculate the above performance measure we may construct a queuing network model. The computational complexity of this model increases very rapidly with system size. Fortunately, as has been shown in [MaGe82], approximate models with a reasonably small error in the final results, can be employed. These are derived by lumping "equivalent" states of the model to obtain a Markov chain of substantially smaller size.

Such a model was derived in [KoPr86] and is shown in Figure 2. In it, at state $(P-i)$ there are $P-i$ processors which are executing their tasks while the remaining i processors are idle being serviced or waiting to be serviced by a memory module.

At a rate of $\beta(i) \cdot \mu$, one of the i idle processors will complete its service, increasing the number of active ones to $P-i+1$. $\beta(i)$ is the average number of processors, out of the i idle ones, that are serviced at a given time instant. Similarly, at a rate of $(P-i)\delta$, an active processor will generate a memory request and join the idle processors, reducing the number of active ones by 1.

To derive an expression for $\beta(i)$, we assume (as in [MaGe82]) that processors request service from the different memory modules with equal probabilities. Hence, the probability that all i requests of the idle processors will be directed to exactly j out of the M memory modules is $Q_{i,j}^{(M)}$ which is defined in (3.9). This probability has to be multiplied by $\min(j,B)$ since only B requests can be serviced if j is greater than the number of busses B.

Finally we obtain,

$$\beta(i) = \sum_{j=1}^{\min(i,M)} Q_{i,j}^{(M)} \min(j,B); \qquad i = 1,2,\dots,P \qquad (4.1)$$

The Markov chain in Figure 2 is a birth and death process, whose solution is easily obtained. Let $\Phi(i)$ denote the steady-state probability of state i, then,

$$\Phi(i) = \frac{\left(\frac{\mu}{\delta}\right)^i \frac{\prod_{k=0}^{i-1} \beta(P-k)}{i!}}{1 + \sum_{m=1}^{P} \left(\frac{\mu}{\delta}\right)^m \frac{\prod_{k=0}^{m-1} \beta(P-k)}{m!}}; \qquad i = 1,2,\dots,P \qquad (4.2)$$

and,

$$\Phi(0) = \frac{1}{1 + \sum_{m=1}^{P} \left(\frac{\mu}{\delta}\right)^m \frac{\prod_{k=0}^{m-1} \beta(P-k)}{m!}}$$

The processing power measure is given by,

$$C(P,M,B) = \sum_{i=1}^{P} i \quad \Phi(i) \qquad (4.3)$$

Expression (4.3) can now be substituted into (3.5) to calculate the expected available processing power.

This calculation was done for a multiple bus multiprocessor VLSI system with the following parameters:

1. Processors with $\alpha = 0.3$, $d = 1.5$, $A = 1$ and $E = 1$ (the area of a processor and its power dissipation are defined to be the unit area and the unit power, respectively), $l = 2$, $p = 0.9$ and $\lambda = \lambda_o$ (time will be measured in $1/\lambda_o$ units). In addition, $\delta/\mu = 0.5$.

2. Memory modules with $\alpha = 0.2$, $d = 1.6$, $A = 0.75$, $E = 0.3$, $l = 2$, $p = 0.92$ and $\lambda = 0.6\lambda_o$.

3.Busses (including the arbiters' circuitry) with $\alpha = 0.12$, $d = 2.0$, $A = 0.25$, $E = 0.1$, $l = 1$, $p = 0.95$ and $\lambda = 0.1\lambda_o$.

For a total chip area of $A = 7.75$ and maximal power dissipation of $E = 6.5$ (measured in units of processor area and processor power dissipation, respectively), three possible systems designs were examined. These systems are $(N_P, N_M, N_B) = (4,4,3)$, $(N_P, N_M, N_D) = (5,3,2)$ and $(N_P, N_M, N_B) = (3,5,4)$, each uses up all the available chip area. The values of the objective function (the expected available processing power) for the three systems in the interval $[0, 0.5/\lambda_o]$ were compared and the results are depicted in Figure 3. From it we can see that for a relatively short mission time the $(5,3,2)$ system is preferable while for a longer mission time the $(4,4,3)$ is the best. The system $(3,5,4)$ is always inferior to the other two with a value of the objective function lower by approximately 13%. If we are concerned only with the yield (which is the percentage of chips having at least 2 processors, 2 memories and one bus which are defect-free) then the order of preference of the above systems is $(4,4,3)$, $(3,5,4)$ and $(5,3,2)$ with yields of 0.226, 0.224 and 0.218, respectively.

If the maximum allowed power dissipation is reduced to $E = 5.3$, the above $(5,3,2)$ and $(4,4,3)$ systems are not feasible any more (their power dissipation exceeds the maximum allowed). However, this does not imply that the $(3,5,4)$ is the best. The $(4,3,4)$ system turns out to be our best choice (see Figure 3) although it does not use up all the available silicon area $(A = 7.25 < 7.75)$. This system has though a yield of 0.215 which is lower than that of the $(3,5,4)$ system (0.224).

We have also checked the possibility of increasing the feature size of the bus to increase its yield. For a total chip area of $A = 8.2$ and a maximal power dissipation of $E = 6.5$, the design yielding the highest value of the objective function is $(4,4,4)$, (note that this system uses only $A = 8$ out of the 8.2 area unit available). If the bus feature size is increased by the factor $\beta = 1.6$, the best possible system is now $(4,4,3)$ (i.e., one bus less) but the achieved value of the objective function is higher by approximately 2.5%. A further increase in the bus feature size does not necessarily have the same effect on the value of the objective function. For example, when increasing the feature size by $\beta = 2$, the optimal system is $(4,4,2)$ for which the value of the objective function is reduced by approximately 0.9% compared to the original $(4,4,4)$ system with $\beta = 1$. The increase in feature size can be as high as $\beta = 2.4$ and still fit into the available chip area. In this case the value of the objective function is higher than the original one by only 0.9% compared to the 2.5% for $\beta = 1.6$. When yields are compared, the $\beta = 1.6$ case is still our best choice with a yield of 0.239 compared to 0.233 and 0.238 for $\beta = 1$ and $\beta = 2.4$, respectively. Other numerical examples where the increase in feature size has even reduced the value of the objective function were observed.

The conclusion that can be drawn from the last examples is that in the wafer-scale design environment, increasing the feature size to reduce the density of circuit failures due to manufacturing defects is not necessarily beneficial.

5. CONCLUSIONS

The problems associated with the design of multiprocessors on large area chips or wafers were presented in this paper. Incorporating fault-tolerance with added redundancy is unavoidable in this environment where several system elements are bound to have defects.

Evaluating the effectiveness of a given redundancy scheme and calculating the optimal values of its parameters are more involved in this design environment where all system elements must share the same silicon area and the same allowed power dissipation. To this end, several appropriate objective functions have been introduced; these may also be used to evaluate and compare different architectures of multi-processors. Such an analysis was illustrated through an example of a multiple bus system.

6. REFERENCES

[AdSi82] G. B. Adams and H. J. Siegel, "The Extra Stage Cube: A Fault-Tolerant Interconnection Network for Supersystems," IEEE Trans. on Computers, Vol. C-31, May 1982, pp. 443-454.

[Fran81] M. A. Franklin, "VLSI Performance Comparison of Banyan and Crossbar Communication Networks," IEEE Trans. on Computers, Vol. C-30, April 1981, pp. 283-291.

[FrWa82] M. A. Franklin, D. F. Wann and W. J. Thomas, "Pin Limitation and Partitioning of VLSI Interconnection Networks," IEEE Trans. on Computers, Vol. C-31, Nov. 1982, pp. 1109-1116.

[Geor84] C.J. Georgiou, "Fault-Tolerant Crosspoint Switching Networks," Proc. of the 14th Symposium on Fault-Tolerant Computing, June 1984, pp. 240-245.

[Kore81] I. Koren, "A Reconfigurable and Fault-tolerant VLSI Multiprocessor Array," Proc. of the 8th Symp. on Comp. Arch., May 1981, pp. 425-441.

[KoBr84] I. Koren and M.A. Breuer, "On Area and Yield Considerations for Fault- Tolerant VLSI Processor Arrays," IEEE Trans. on Computers, Vol. C-33, Jan. 1984, pp. 21-27.

[KoPr85] I. Koren and D.K. Pradhan, "Introducing Redundancy into VLSI Designs for Yield and Performance Enhancement," Proc. of the 15th Annual Symposium on Fault-Tolerant Computing, June 1985, pp. 330-335.

[KoPr86] I. Koren and D.K. Pradhan, "Modeling the Effect of Redundancy on Yield and Performance of VLSI Systems," to appear, IEEE Trans. on Computers.

[KuRe85] V. P. Kumar and S. M. Reddy, "Design and Analysis of Fault-Tolerant Multistage Interconnection Networks with Low Link Complexity," Proc. of the 12-th Annual Symp. on Comp. Arch., June 1985, pp. 376-386.

[LaVa82] T. Lang, M. Valero and I. Alegre, "Bandwidth of Crossbar and Multiple-Bus Connections for Multiprocessors," IEEE Trans. on Computers, Vol. C-31, December 1982, pp. 1227-1234.

[MaGe82] M. Ajmone Marsan and M. Gerla, "Markov Models for Multiple Bus Multiprocessor Systems," IEEE Trans. on Computers, Vol. C-31, March 1982, pp. 239-248.

[PaLa83] K. Padmanabhan and D. H. Lawrie, "A Class of Redundant Path Multistage Interconnection Networks," IEEE Trans. on Computers, Vol. C-32, Dec. 1983, pp. 1099-1108.

[Prad83] D.K. Pradhan, "Fault-tolerant Architectures for Multiprocessors and VLSI Systems", Proc. of the 13th Symp. on Fault-Tolerant Computing, June 83.

[SeAg84] S.C. Seth and V.D. Agrawal, "Characterizing the LSI Yield Equation from Wafer Test Data," IEEE Trans. on Computer-Aided Design, Vol. CAD-3, April 1984, pp. 123-126.

[StMc80] C.H. Stapper, A.N. McLaren and M. Dreckman, "Yield Model for Productivity Optimization of VLSI Memory Chips with Redundancy and Partially Good Product," IBM J. Res., Dev., Vol. 24, No. 3, May 1980, pp. 398-409.

[StAr83] C.H. Stapper, F.M. Armstrong and K. Saji, "Integrated Circuit Yield Statistics," Proc. of IEEE, Vol. 71, April 1983, pp. 453-470.

[Stap84] C.H. Stapper, "Modeling of Defects in Integrated Circuits Photolithographic Patterns," IBM J. Res., Dev., Vol. 28, No. 4, July 1984, pp. 461-475.

[TzYe85] N. Tzeng, P. Yew and C. Zhu, "A Fault-Tolerant Scheme for Multistage Interconnection Networks," Proc. of the 12-th Annual Symp. on Comp. Arch., June 1985, pp. 368-375.

ISRAEL KOREN received the B.Sc. (Cum Laude), M.Sc. and D.Sc. degrees from the Technion - Israel Institute of Technology, Haifa, in 1967, 1970, and 1975, respectively, all in Electrical Engineering.

Since 1979 he is with the Departments of Electrical Engineering and Computer Science at the Technion - Israel Institute of Tech- nology, where he became the Head of the VLSI Systems Research Center in 1985. Previously he has held positions with the University of California, Santa Barbara and the University of Southern California, Los Angeles. In 1982 he was on sabbatical leave with the University of California, Berkeley.

He has been a consultant to National Semiconductor, Israel, in architecture of microprocessors and high-speed algorithms for arithmetic operations, in 1984-1986, to Tolerant Systems, San Jose, California, in architecture of fault-tolerant distributed computer systems in 1983, and to ELTA, Electronics Industries, Israel, in architecture of parallel signal processors in 1981-1982.

Currently he is a Visiting Professor at the University of Massachusetts, Amherst. Dr. I. Koren's current research interests are Fault-Tolerant VLSI and WSI Architectures, Models for Yield and Performance, Reliability Evaluation of Computer Systems and Computer Arithmetic.

ZAHAVA KOREN received the B.A and M.A degrees in Mathematics and Statistics from the Hebrew University in Jerusalem in 1967 and 1969, respectively, and the D.Sc. degree in Operations Research from the Technion - Israel Institute of Technology in 1976.

From 1967 to 1968 she was a Teaching assistant in the Dept. of Statistics, Hebrew University in Jerusalem. From 1969 to 1974 she was a Teaching Instructor in the Dept. of Industrial Engineering at the Technion, and from 1975 to 1976 a Teaching Instructor in the Dept. of Statistics, University of Haifa, lecturing Statistical Inference, Queuing Theory, Inventory Theory, and Stochastic Processes. From 1972 to 1976 she was a Consulting Statistician in medical and psychological experiments performed at the Medical School, Tel-Aviv University. In 1979 she was an Assistant Professor in the Dept. of Business and Economics, California State University in Los Angeles, lecturing Statistics and Operations Research. From 1980 to 1984 she was a Lecturer in the Dept. of Statistics, University of Haifa, and a Consultant to SHAHAF, a public opinion research institute. Since 1985 she is with the Dept. of Computer Science at the Technion, Haifa.

Dr. Z. Koren's main interests are Optimization in Queuing Processes, Stochastic Processes and Reliability.

DHIRAJ K. PRADHAN is currently a Professor in the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst. Previously, he has held positions with the University of Regina, Sask. Canada, Oakland University, Rochester, MI, Stanford University, Stanford, CA, and IBM Corporation. He has been actively involved with research in fault- tolerant computing and parallel processing since receiving his Ph.D. in 1972. He has presented numerous papers on fault-tolerant computing and parallel processing. He has also published extensively in journals such as *IEEE Transactions* and *Networks*. His research interests include fault- tolerant computing, computer architecture, graph theory, and flow networks.

Dr. Pradhan has edited the special issue on Fault-Tolerant Computing, published in *IEEE Transactions on Computers* (April 1986) and *IEEE Computer*, (March 1980). Also he has served as Session Chairman and Program Committee Member for various conferences. Currently he is an Editor for the *Journal of VLSI and Digital Systems*. He is also the editor and coauthor of the book entitled, *Fault-Tolerant Computing: Theory and Techniques*, Vol. I and II, published by Prentice-Hall.

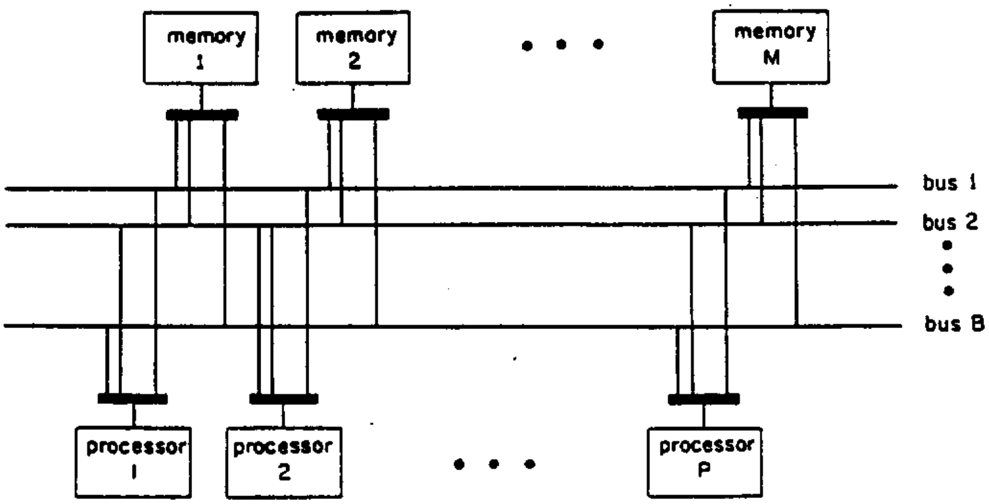

Fig. 1: A P*M*B multiple bus multiprocessor

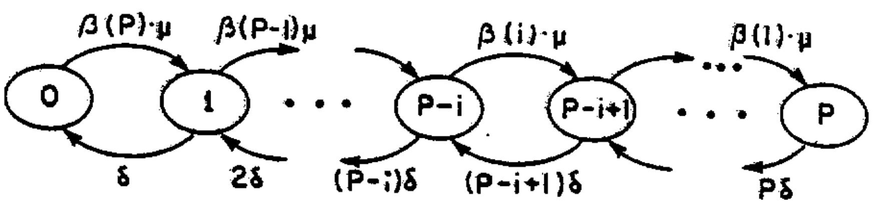

Fig. 2: A birth and death model for a P*M*B multiple bus multiprocessor

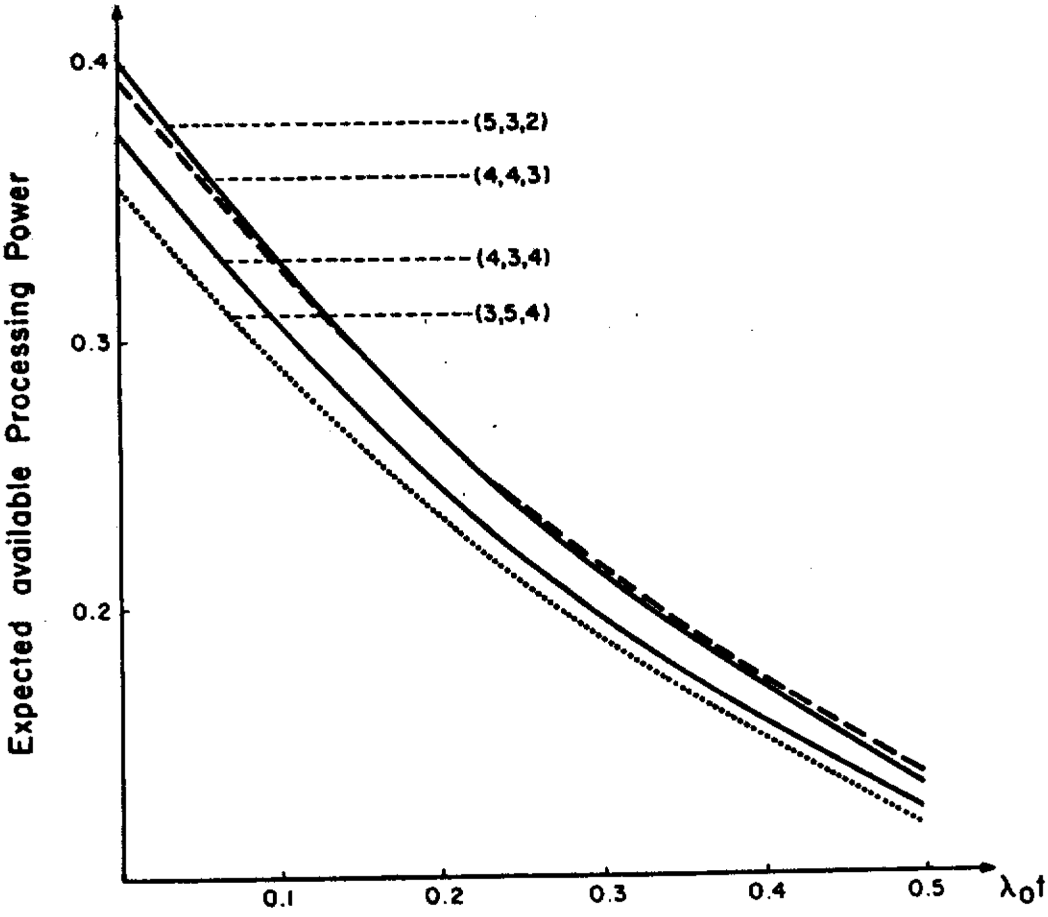

Fig. 3: The expected available processing power for different (N_P, N_M, N_B) multiple bus systems.