Reconfiguration and Analysis of a Fault-Tolerant Circular Butterfly Parallel System

Nian-Feng Tzeng, *Senior Member, IEEE*

Abstract— **The butterfly parallel system has a regular and simple interconnection pattern, making it suitable for VLSI/WSI implementation. In this paper, we propose an effective fault-tolerant technique for the circular butterfly parallel system to ensure its rigid full butterfly structure even in the presence of failures, addressing reconfiguration in detail. A resulting butterfly system with L levels involves $(1/\log_2 L)\%$ spare PE's and approximately 50% additional links. The reconfiguration process of our design in response to any operational fault is easy and can be performed in a distributed manner. The reliability and layout of this proposed design are evaluated analytically. This design, due to its specific configuration, exhibits significantly improved reliability while taking only moderate extra layout area.**

Index Terms—**Butterfly parallel systems, fault-tolerance, reconfiguration process, reliability evaluation, VLSI layout.**

I. Introduction

As a result of the recent advents of VLSI and WSI (wafer scale integration), it is feasible to construct a parallel system comprising a large number of interconnected processing elements (PE's) fabricated on a single chip or wafer. A butterfly-based network architecture is considered very promising among the high-performance parallel systems, because it is capable of efficiently solving a wide variety of algorithms [1], [2] and may also be tailored to special purpose applications [3]. The butterfly network architecture involves a simple interconnection pattern, and its every PE has only four ports, irrespective of the system size. Consequently, an area-efficient layout for the butterfly parallel system is realizable, making it readily suited for VLSI/WSI implementation.

One fundamental consideration in designing a parallel system is its reliability, and system reliability clearly becomes more important with its increasing use in life critical applications and in environments where system failures can cause significant economic loss. As the number of PE's in a system grows, its reliability tends to drop rapidly, unless a proper fault-tolerant technique is incorporated in the system design to guard against operational failures.[1] In general, a fault-tolerant system responds to operational failures by reconfiguring itself to exclude failed components, while keeping all nonfaulty components whenever possible. A system, after reconfiguration, is often required to maintain the same topology and size, if it is not a gracefully degradable system. This is particularly important for high-performance parallel architecture, like the butterfly system, because tasks to be executed are partitioned in a way that best matches the PE interconnection pattern of the target machine, and any change in its topology or size tends to jeopardize execution efficiency considerably. A high-performance parallel system thus needs redundant PE's and links to ensure its topology and size remain unchanged even in the presence of operational failures.

Though essential, the fault-tolerant issue of the butterfly parallel system has been treated inadequately. This paper concerns a technique able to significantly improve the reliability of butterfly parallel systems, and the resulting system is more reliable than a previous design. In our design, the reconfiguration process in response to an operational failure is simple and can be carried out in a distributed manner. More importantly, the system retains the rigid butterfly topology after reconfiguration, ensuring the same high performance. Our design takes insignificant extra area (often less than 50%) under practical layout situations where PE width is much larger than link width.

A butterfly parallel system with size $L \times \log_2 L$ involves $\log_2 L$ stages each with L PE's, and has the butterfly connection pattern among PE's in consecutive stages. The PE's in the last stage are also connected to the PE's in the first stage through the butterfly wrap-around connection, forming a *c*ircular *b*utterfly *p*arallel *s*ystem (CBPS). A CBPS with 24 PE's (i.e., $L = 8$) is shown in Fig. 1, where the first stage is repeated at the right end for the simplicity of illustration. Every PE has, in addition to local memory, logics for sending, receiving, and forwarding messages. The CBPS is similar in organization to the loop-structured switching network [1], the MAN-YO network architecture [3], the chained-structured butterfly architecture [2], and the Lambda network addressed recently [4]. A small scale MAN-YO system was prototyped at NEC Corporation [3].

Notice that the butterfly parallel system under consideration here is different from a butterfly network used in the switching context where the network is for interconnecting nodes on its side(s) (and essentially belongs to the class of multistage interconnection networks [5]). Each node in a butterfly parallel system involves the processing unit, storage, and communication logics, whereas a node in a butterfly interconnection network is simply a switch only. A butterfly parallel system directly connects its constituent nodes together in accordance with the butterfly topology, offering good scalability and cost-effectiveness.

Manuscript received January 5, 1991; revised December 10, 1991. This work was supported by the NSF under Grant MIP-8807761.

The author is with The Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504.

IEEE Log Number 9210513.

[1] By an operational failure, we mean any incorrect operation of a hardware component due to a fault within the component.

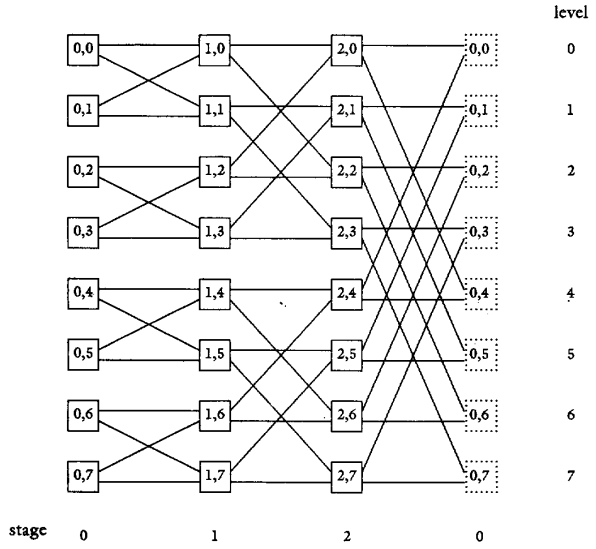

Fig. 1. A circular butterfly parallel system with 24 PE's. (Stage 0 is duplicated for easy illustration.)

There are a few research reports on fault-tolerant butterfly interconnection networks [14]–[16], where the fault-tolerant issue is to assure the existence of a path between every network input and output in the presence of faults, which can be achieved without maintaining the rigid network topology. Their design techniques cannot apply to the butterfly parallel system so as to guarantee its strict full butterfly topology desired after faults arise, and thus their results will not be compared with ours.

In Section II, the CBPS topology is briefly reviewed. In Section III, the proposed CBPS design is introduced, and then the reconfiguration procedure for such a design is given. Section IV evaluates the reliability of the proposed CBPS. The layout of this design is considered in Section V. Concluding remarks appear in Section VI.

II. Review of the CBPS

The CBPS of size $N = L \times \log_2 L$ is organized as L levels of $\log_2 L$ stages. Stages in the CBPS are numbered rightwards in a sequence from 0 to $(\log_2 L - 1)$ with 0 for the leftmost stage. Similarly, the PE's in every stage are labeled downwards from 0 to $L - 1$, called their *level* numbers, with 0 for the top PE. A PE at the jth level of the ith stage in the CBPS can be uniquely denoted by <i, j>.

Every link in the CBPS is unidirectional (heading toward the right), and each PE has two outgoing and two incoming links. Formally, the topology describing rules of the CBPS are given below:

1) The upper outgoing link of PE <i, j> is connected to PE <$i + 1 \pmod{\log_2 L}$, $j - 2^i \times l_i$>, and
2) The lower outgoing link of PE <i, j> is connected to PE <$i + 1 \pmod{\log_2 L}$, $j + 2^i \times (1 - l_i)$>,

where l_i is the bit with weight 2^i in the binary representation of j $(= l_{\log_2 L-1} \cdots l_1 l_0)$.

A PE at stage k, level V $(= l_{\log_2 L-1} \cdots l_{k+1} l_k l_{k-1} \cdots l_0)$: PE <$k$, $l_{\log_2 L-1} \cdots l_{k+1} l_k l_{k-1} \cdots l_0$> is connected through its upper outgoing link to PE <$k + 1 \pmod{\log_2 L}$, $l_{\log_2 L-1} \cdots l_{k+1} 0 l_{k-1} \cdots l_0$> and through its lower outgoing link to PE <$k+1 \pmod{\log_2 L}$, $l_{\log_2 L-1} \cdots l_{k+1} 1 l_{k-1} \cdots l_0$>. Traversing the upper outgoing link of any PE reaches a PE whose level address has its kth bit equal to 0, irrespective of its original value; likewise, the lower outgoing link leads messages to a PE with its level address having bit k to be 1. In other words, the two outgoing links of this PE connect two PE's which are at level $(\mathrm{V}-2^k \times l_k)$ and level $(\mathrm{V}+2^k \times (1 - l_k))$, respectively; these two PE's are referred to as the *right-connected PE's* of this PE.

Two PE's are called *conjugate PE's* with each other if they have identical right-connected PE's. In Fig. 1, for example, PE <1, 1> and PE <1, 3> are conjugate PE's since they share the same two right-connected PE's, PE <2, 1> and PE <2, 3>. The concept of conjugate PE's is essential to our fault-tolerant CBPS design discussed subsequently.

While we assume links in the CBPS to be unidirectional, it should be noted that the same reliable system can result when links are designed to be bidirectional. For the CBPS with bidirectional links, the wrap-around connections between the two end stages are no longer needed. However, the routing procedure then becomes more complicated and the number of pins required per PE is larger, because a bidirectional link either consists of two undirectional links, one along each direction, or needs more control signals.

III. Proposed Design

A. Design Methodology

The basic idea of our design is to add one extra PE per level, plus appropriate extra links among PE's so that a failed PE can be replaced by its immediate right PE. The proposed CBPS with 24 PE's and 8 spares (i.e., $L = 8$) is depicted in Fig. 2. (As in Fig. 1, stage 0 is replicated in this figure for easy illustration.) Every PE in the design has at most six links and the spare PE of every level is placed to the right of stage 2 (i.e., between stages 2 and 0), with the wrap-around connections now existing between the "column" of spares (called stage s) and stage 0. The additional links and spare PE's are denoted by dashed lines, while the solid lines represent the original CBPS. Like the regular links, the additional links are also unidirectional. (Every PE in our design is assumed to be equipped with six links for the sake of regularity.)

Consider the case where PE <1, 1> fails in Fig. 2. The faulty PE is bypassed upon reconfiguration, and then its two right-connected PE's, PE <2, 1> and PE <2, 3>, are reconfigured in such a way that the added links originating from them are activated while the two outgoing links connected respectively to PE <s, 5> and PE <s, 7> are deactivated, effectively "emulating" the connection pattern between level 1 and level 3 present in the immediate earlier stage. After PE <2, 1> and PE <2, 3> are reconfigured, their right-connected PE's, namely, PE <s, 1>, PE <s, 5>, PE <s, 3>, and PE <s, 7>, are also reconfigured so as to realize the appropriate butterfly connection pattern existing in the immediate earlier stage.

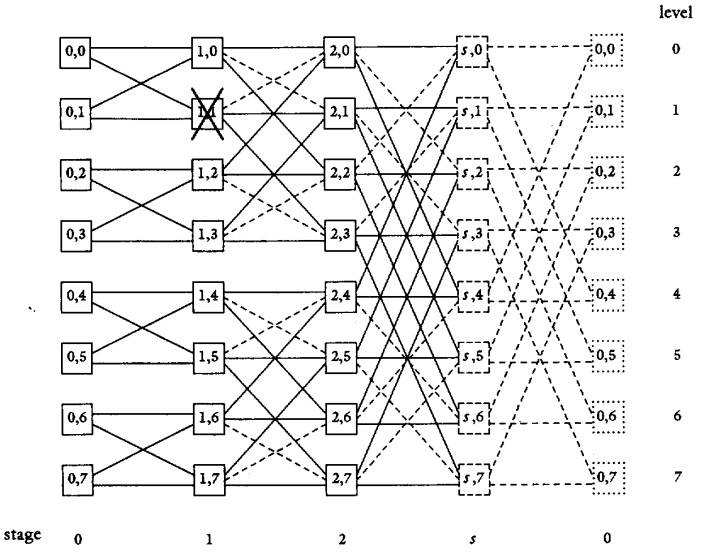

Fig. 2. The fault-tolerant CBPS obtained from Fig. 1 by adding extra links and spare PE's. (Extra links and spares are denoted by dashed lines. Stage 0 is replicated for easy illustration.)

More precisely, the outgoing links from PE <s, 1> to PE <0, 5>, from PE <s, 5> to PE <0, 1>, from PE <s, 3> to PE <0, 7>, and from PE <s, 7> to PE <0, 3> are activated. The remaining four PE's in stage s are bypassed to preserve the butterfly topology. Interestingly, when a PE is bypassed or reconfigured, its conjugate PE is also bypassed, *regardless of whether it is faulty or not,* so as to elegantly maintain the rigid butterfly topology. In the presence of failed PE <1, 1>, its conjugate PE (i.e., PE <1, 3>), despite being fault-free, is bypassed so that PE <2, 1> and PE <2, 3> become the logical successors respectively of PE <0, 1> and PE <0, 3> along level 1 and level 3. Likewise, since PE <2, 1> and PE <2, 3> are reconfigured, their conjugate PE's, PE <2, 5> and PE <2, 7>, are bypassed upon reconfiguration as well. The CBPS so reconfigured is shown in Fig. 3, where solid lines denote active links and dotted lines are inactive ones.

Here, every PE is assumed to have four control switches that allow the PE to be bypassed, as illustrated in Fig. 4(a). If a PE is to be bypassed, the four switches are set as shown in Fig. 4(b); otherwise, the four switches are set oppositely. (Employing control switches to bypass a failed/unused component can also be found in other designs [6], [7].)

Bypassing the faulty PE <1, 1> renders PE <2, 1> the logical successor to PE <0, 1> along level 1 in Fig. 2. To retain the butterfly topology, the two incoming links of PE <1, 1> must be extended through two dedicated wires to PE <2, 1>. One dedicated wire between PE <1, 1> and PE <2, 1> already exists. The second wire can be realized by making use of the added links between PE <1, 1> and PE <2, 0> and between PE <1, 0> and PE <2, 1> plus one control switch, as shown in Fig. 4(c). When the switch is set at the "V" state that connects the upper input to the upper output and the lower input to the lower output [illustrated in Fig. 4(d)], a second dedicated wire is made between PE <1, 1> and PE <2, 1> (and also between PE <1, 0> and PE <2, 0>). On the other hand, when

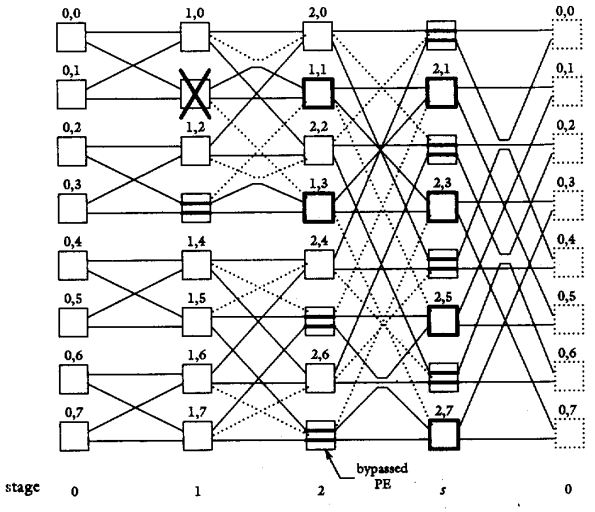

Fig. 3. A reconfigured CBPS in the presence of a failed PE as shown. (Solid lines indicate active links after reconfiguration. A bypassed PE has two bold lines across it, and a PE at the "emulating" state is darkened. The numbers show the PE logic addresses, rather than physical ones.)

the switch is set in an opposite manner, called the "X" state [see Fig. 4(d)], dedicated wires are present between PE <1, 1> and PE <2, 0> and between PE <1, 0> and PE <2, 1>. If a PE is bypassed, the control switch on its added outgoing link is set at the "V" state so as to make an extra dedicated wire to the right neighbor; whereas if a PE is reconfigured, the switch is set at the "X" state to emulate the "cross" connection style existing in the earlier stage. Layout area needed for the extra links and associated control switches is insignificant, as will be seen in Section V.

Links and switches are considered to be far more reliable than PE's, as revealed by previous studies [12], since the

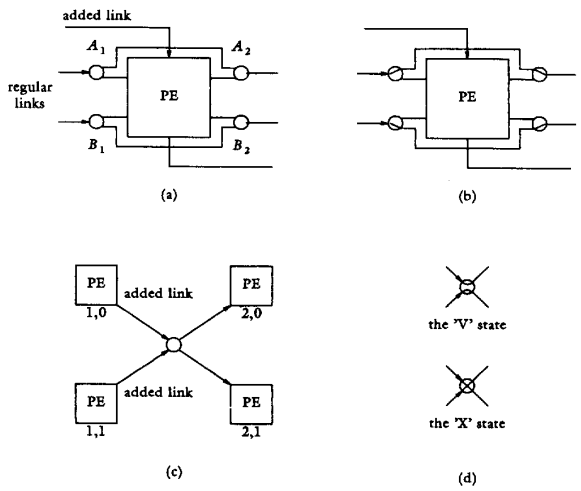

Fig. 4. Control switches needed for reconfiguration.

PE is relatively much more complicated. The number of components could be tens or even hundreds of times as many in a typical PE as in a switch. For example, a 17-bit switch plus interfaces contains some 1266 transistors in the design given in [13], whereas a PE normally involves tens, if not hundreds, of thousands of transistors. Failures at certain links may be viewed as PE failures. More specifically, the failure of a regular outgoing link of a PE can be treated as the PE failure and is tolerable, provided that the link connects a PE located at a different level. As an instance, the failure of the link between PE <1, 1> and PE <2, 3> in Fig. 2 can be treated as the failure of PE <1, 1> and is thus tolerable. Other links which connect PE's in the same level are expected to be very reliable because they are shorter and often involve no off-chip connections (as one or several levels of PE's can be fabricated within a chip/wafer; this horizontally partitioning fabrication is more pin-efficient than a vertically partitioning fabrication because a link connecting two PE's in the same level within a chip requires no pin). In addition, we may add some redundant wires to those intolerant links, and they will be trivially switched in, upon the occurrence of failed wires [8]. Thus, failure rates of links and control switches are assumed to be negligibly low, and no attempt is made to consider the failure at each link or switch individually. Instead, all the links/switches in a system will be treated as a whole, with a "lump" failure rate assigned to them in our later reliability analysis.

B. Resulting CBPS

Consider a CBPS with L levels. A spare PE is added to the right of stage $\log_2 L - 1$ at each level. Generally, the additional links between stages i and $i+1 \pmod{\log_2 L + 1}$, $1 \le i \le \log_2 L$ ($= s$, namely, stage s is treated as stage $\log_2 L$), are introduced in such a way that they emulate the connection pattern between stages $i-1$ and i (notice that the wrap-around connection pattern between stages s and 0 is the same as that between stages $\log_2 L - 1$ and 0 of the original CBPS). The interconnection of the additional links in the proposed CBPS is formally characterized as follows.

1) The additional outgoing link of PE <i, j> is connected to the additional incoming link of PE <$i+1 \pmod{s+1}$, $j + (-1)^{l_{i-1}} 2^{i-1}$> for $1 \le i \le s$ ($= \log_2 L$), and
2) The upper (or lower) regular outgoing link of PE <s, j> is connected to a regular incoming link of PE <0, j> for $l_{\log_2 L - 1}$ equal to 0 (or 1),

where l_i is the bit with weight 2^i in the binary representation of j. Also, additional connections plus switches between PE <0, j> and PE <1, $j + (-1)^{l_0}$> in stages 0 and 1 are in place of the regular connections between them so that a PE in stage 0 can be "bypassed." The resulting CBPS with $L = 8$ is illustrated in Fig. 2. Every PE in stage 0 leaves one regular incoming link and one regular outgoing link unused, whereas every PE in stage 1 leaves one regular incoming link unused and every spare PE has one unused regular outgoing link. Consequently, every level has two unused incoming and two unused outgoing links. These unused links may be employed for I/O purposes, and every level is guaranteed to have at least one pair of workable I/O ports after reconfiguration. This is because at each level the two incoming links are associated with distinct PE's and so are the two outgoing links.

Making additional connections this way achieves simultaneously the following desired goals: i) low area overhead taken by the extra links and control switches (as will be seen later), ii) avoiding long interconnection wires because a faulty PE is replaced by its immediate right neighbor rather than a distant spare, and iii) a reasonably simple reconfiguration procedure which can be performed in a distributed manner. An impressive reliability enhancement results from this design.

C. Reconfiguration Procedure

Suppose that a PE, say PE <i, j>, fails in the proposed CBPS with L levels, $0 \le i < s$ (the failure at stage s requires no reconfiguration since spares are bypassed normally). The reconfiguration process works from stage i on, stage by stage rightwards, until stage s is encountered and necessary spares are brought into the system. A PE can be activated or deactivated, depending upon the settings of its control switches given in Fig. 4(a). A failed PE or an unused PE is deactivated by setting the switches as illustrated in Fig. 4(b). Such a deactivated PE is termed at the "bypassed" state. On the other hand, a PE in the system gets activated by setting all the switches oppositely. An activated PE can be at the "regular" state (if its both regular outgoing links are active and the additional outgoing link is inactive) or at the "emulating" state (if the additional outgoing link is active and any one of its two regular outgoing links is active).

Initially, all the PE's are set to the "regular" state, with all the spare PE's "bypassed." Once the said faulty PE arises and is identified, the reconfiguration process removes the PE from the system by setting it to the "bypassed" state, and then its conjugate PE is also set to the "bypassed" state. At stage $i+1$, exactly two PE's (i.e., the two right-connected PE's of faulty PE <i, j>) are set to the "emulating" state; the two PE's are located at level $j - 2^i \times l_i$ and level $j + 2^i \times (1 - l_i)$. The

conjugate PE's of the two PE's are then set to the "bypassed" state. In general, at stage k ($i < k \leq s$), there are 2^{k-i} specific PE's set to the "emulating" state, and their locations can be readily pinpointed from earlier stages iteratively since a PE at the "emulating" state causes its both right-connected PE's to be set at the "emulating" state. Then the conjugate PE's of those 2^{k-i} specific PE's in stage k are set to the "bypassed" state.

Consider the failure of PE <1, 1> shown in Fig. 2 as an example. The failed PE and its conjugate PE, PE <1, 3>, are both set to the "bypassed" state. PE's at the same levels as the two bypassed PE's in the next stage, i.e., PE <2, 1> and PE <2, 3>, are then set to the "emulating" state, with their conjugate PE's, PE <2, 5> and PE <2, 7>, set to the "bypassed" state. Finally, four PE's in stage s, PE <s, 1>, PE <s, 3>, PE <s, 5>, PE <s, 7>, are set to the "emulating" state. The result after reconfiguration due to this failure is depicted in Fig. 3.

The selection of the proper output ports at each PE when its state changes is straightforward. When the PE is set at the "bypassed" or "emulating" state, one of its two regular output ports remains active, with the other regular one disabled and the extra output port enabled. The active regular link is the one that connects to the immediate right PE at the same level. Specifically, for PE <k, β> at the "bypassed" or "emulating" state in stage k, $i \leq k < \log_2 L$, its upper regular outgoing link is active if the bit with weight 2^k in the binary representation of β is 0; otherwise, its lower regular outgoing link is active. As mentioned earlier, a PE at the "emulating" (or "bypassed") state sets the control switch on its extra outgoing link to the "X" (or "V") state.

The reconfiguration process can proceed in a distributed manner if i) the PE is self-testable, allowing it to determine its own status [9], ii) every PE can read the (setting) state of any of its directly right-connected PE's, and iii) every PE has two specific locations upp_left_ok and low_left_ok associated respectively with the upper and lower incoming ports; location upp_left_ok (or low_left_ok) can be set by the PE that connects this PE through the upper (or lower) incoming link. Before the process starts, every PE performs a self-test individually and resets its own upp_left_ok and low_left_ok. Then, the following two-phase process is carried out by PE's, with Phase 1 for setting PE's (if any) to the "emulating" state and Phase 2 for setting PE's to the "bypassed" state. Every healthy PE in stage i, $0 \leq i \leq \log_2 L$, performs Phase 1, stage by stage rightwards, starting with stage 0:

Phase 1

1) if a PE finds both its upp_left_ok *and* low_left_ok set, its state is unchanged; otherwise, it is set to the "emulating" state; and
2) if a PE is at the "regular" state, it sets the location upp_left_ok or low_left_ok at each of its two right-connected PE's; otherwise, it does not.

Notice that PE's in stage 0 perform only Step 2), whereas PE's in stage $\log_2 L$ perform only Step 1). Every healthy PE in stage i, $0 \leq i < \log_2 L$, performs Phase 2, stage by stage rightwards, starting with stage 0; Phase 2 may start after PE's in stage 1 finish Phase 1 operations. Phase 2 results from the fact that once the PE's at the "emulating" state in stage $k+1$ are known, those PE's to be set at the "bypassed" state in stage k can also be identified.

Phase 2

1) every PE reads the (setting) states of its two right-connected PE's; and
2) if a PE finds both read states equal to the "emulating" state, the PE is set to the "bypassed" state; otherwise, its state remains unchanged.

Finally, all the unused spare PE's (i.e., PE's at stage $\log_2 L$ which are not at the "emulating" state) are set to the "bypassed" state. The reconfigured system in the presence of failed PE <1, 1> in CBPS with $L = 8$ is shown in Fig. 3. Any PE has two active outgoing links and two active incoming links, despite being equipped with three outgoing and three incoming links. Every level now contains exactly three active PE's, regardless of whether it involves the failed PE or not.

D. System Properties

The proposed design with L levels contains L spare PE's and $L \times (\log_2 L + 1)$ additional links. As a result, the PE overhead ratio and the link overhead ratio equal $1/\log_2 L$ and $(\log_2 L + 1)/(2\log_2 L)$, respectively. This design can sustain multiple faulty PE's, provided that they satisfy certain constraints as will be discussed in the following section. A reconfigured system maintains the rigid butterfly topology.

Another fault-tolerant design, known as the reconfigurable chain-structured butterfly architecture (RECBAR), was introduced previously [2]. The RECBAR is formed by adding an extra level to the original CBPS and by making additional connections between PE's in different levels. Each PE has three input and output ports except those in the first and last stages, which have two input and three output ports, and three input and two output ports, respectively. The topology describing rules of the RECBAR are as follows: 1) the upper outgoing link of PE <i, j> is connected to PE <$i+1 \pmod{\log_2 L}$, $j - 2^i \pmod{(L+1)}$>; 2) the middle outgoing link of PE <i, j> is connected to PE <$i+1 \pmod{\log_2 L}$, j>; and 3) the lower outgoing link of PE <i, j> is connected to PE <$i+1 \pmod{\log_2 L}$, $j + 2^i \pmod{(L+1)}$>. The RECBAR involves a PE overhead ratio of $O((\log_2 N)/N)$ and a link overhead ratio of approximately 50%. Our design has a larger number of spare PE's but is also more reliable when compared with the RECBAR.

IV. Reliability Evaluation

We first examine what kinds of multiple faults are tolerable in the proposed CBPS with L levels. When a fault arises in stage i, $0 \leq i < s\ (= \log_2 L)$, according to the reconfiguration process, totally δ_1 PE's are set to the "emulating" state, where $\delta_1 = \sum_{j=i+1}^{s} 2^{j-i} = 2^{s-i+1} - 2$. If any subsequent fault happens to these δ_1 PE's, then it is intolerable and the CBPS fails. Moreover, to keep the system operational, a subsequent fault cannot occur at certain PE's in any stage j, for $j < i$, as well. In Fig. 5, for example, the failure of PE <2, 5> also excludes any subsequent fault being at PE <1, 5> and PE <1, 7> in stage 1; and being at the four lower PE's in stage 0

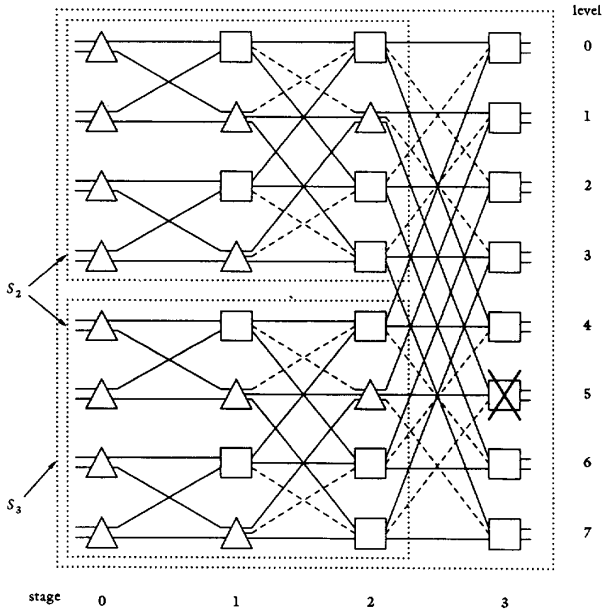

Fig. 5. The structure of S_3 comprising three parts: B_3 and two S_2's. (The wrap-around butterfly connections are omitted for clarity.)

(indicated by triangles). This is because any such subsequent fault, on reconfiguration, requires PE <2, 5> to be set at the "emulating" state, a surely unsuccessful attempt. Note that all the triangles and squares in this figure denote identical PE's, and they are so drawn simply to facilitate later discussion. In general, a fault in stage i also disallows totally δ_2 PE's in stages earlier than i to become faulty, where $\delta_2 = \sum_{j=0}^{i-1} 2^{j+1} = 2^{i+1} - 2$. These δ_2 PE's are referred to as *critical-E* PEs; while those δ_1 PE's are as *critical-L* PE's (where E and L stand for *e*arlier and *l*ater, respectively). The union of critical-E PE's and critical-L PE's are called *critical* PE's.

Similar to critical-L PE's, critical-E PE's can also be identified stage by stage, but from stage $i-1$ leftwards. Apparently, in the presence of the said fault, any subsequent fault, say fault b, in the designed CBPS is tolerable, *as long as it is not a critical PE.* After fault b arises, the critical PE's are redetermined, and the number of the critical PE's now depends upon the positions of the two existing failed PE's. A third fault is tolerable, provided that it does not belong to the set of the newly determined critical PE's. Similarly, this criterion can be employed to decide whether an arising fault is tolerable, given any number of faults present in the system. Our reliability analysis is based on this idea.

We make the following assumptions to facilitate reliability evaluation: i) the PE becomes faulty randomly and independently, with a constant failure rate λ, and ii) links and control switches are reliable and thereby are not treated individually; instead, a "lump" failure rate is assigned to all the links and switches. A system is considered *failed* when the number and the locations of the faulty PE's prevent it from reconfiguring back to its original form and size. Let $Q_s(k)$ be the probability that the designed CBPS with $L = 2^s$ levels can still function after having k faulty PE's. The reliability of the CBPS is then given by

$$R_{CBPS,s}(t) = \sum_{k=0}^{N_s} Q_s(k) \times \binom{N_s}{k} \times \left(e^{-\lambda t}\right)^{N_s-k} \times \left(1 - e^{-\lambda t}\right)^k \quad (1)$$

where N_s is the total number of PE's in the proposed CBPS.

To calculate $Q_s(k)$, we adopt a recursive approach. As shown in Fig. 5, the CBPS with 2^3 levels contains three parts: a set of eight PE's interconnected by the butterfly connecting pattern and two CBPS's each with 2^2 levels. Let the structure of the CBPS with 2^l levels be denoted by S_l; then, in general S_j can be viewed as consisting of three parts: a set of 2^j PE's interconnected by an appropriate butterfly connecting pattern, represented by B_j, and two S_{j-1}'s (i.e., CBPS's each with 2^{j-1} levels). The calculation of $Q_j(k)$ depends not only on the faults (and their positions) in B_j but also on the faults (and their positions) in the two S_{j-1}'s. For arbitrary failed PE's existing in B_j, any subsequent failure would not cause S_j to fail, if it does not belong to the set of critical PE's (as stated previously). It is easy to validate that in the presence of arbitrary faults in B_j, one half of the critical-E PE's fall into one S_{j-1} and the other half fall into the second S_{j-1}. When PE <3, 5> in Fig. 5 fails, for example, the critical-E PE's are those denoted by triangles in the two S_2's. We use $\eta_j(i_{j1})$ to denote the number of noncritical-E PE's in S_{j-1}, when i_{j1} faults, $0 \le i_{j1} \le k$, are present in B_j. Let Ξ_{j2} be the number of intolerable fault patterns while distributing i_{j2} faults into those $\eta_j(i_{j1})$ noncritical-E PE's in the first S_{j-1}; and Ξ_{j3} be the number of intolerable fault patterns while distributing i_{j3} faults into those noncritical-E PE's in the second S_{j-1}, where $i_{j1} + i_{j2} + i_{j3} = k$. Then, we have $q_j(i_{j1})$, the probability that B_j, in the presence of i_{j1} faults, does not corrupt S_j, given that there are i_{j2} and i_{j3} faults respectively in the two S_{j-1}'s, as follows (recall that N_{j-1} denotes the total number of PE's in S_{j-1}):

$$q_j(i_{j1}) = \frac{\binom{\eta_j(i_{j1})}{i_{j2}} - \Xi_{j2}}{\binom{N_{j-1}}{i_{j2}}} \times \frac{\binom{\eta_j(i_{j1})}{i_{j3}} - \Xi_{j3}}{\binom{N_{j-1}}{i_{j3}}}. \quad (2)$$

It is fairly involved to calculate exact $\eta_j(i_{j1})$, Ξ_{j2}, and Ξ_{j3} analytically, because they depend on both the number and the locations of faults. Instead, we derive a lower bound and an upper bound on $q_j(i_{j1})$. Their derivation can be found in Appendix. With the two bounds on $q_j(i_{j1})$, we can obtain the bounds on $Q_j(k)$ by summing up the probabilities of all the tolerable fault patterns with i_{j1}, i_{j2}, and i_{j3} failed PE's existing respectively in the three parts of S_j, for all $i_{j1} + i_{j2} + i_{j3} = k$. Using (1), the two bounds on $R_{CBPS,s}(t)$ are derived. A program is written to calculate the two bounds of any given CBPS.

In Fig. 6, the lower and upper reliability bounds on the CBPS with 2^4 levels and with 2^8 levels are plotted as a function of time, where the PE failure rate λ is assumed to be 1.0 per unit time (e.g., 10^6 h) and the "lump" link and switch failure rate is $s\lambda$ for a system with 2^s levels. These two

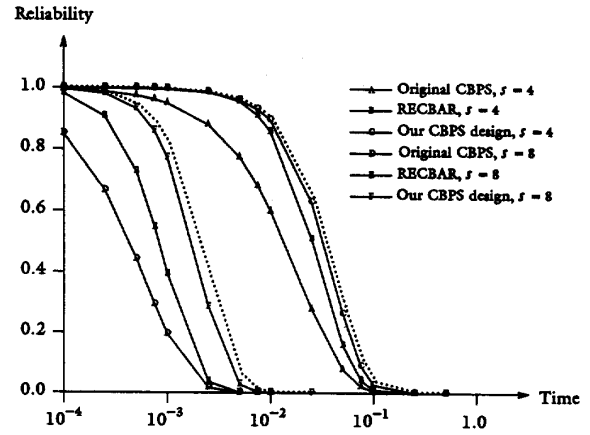

Fig. 6. Reliability comparison among the original CBPS, the RECBAR, and our CBPS design. (The dotted curves show the reliability upper bounds of our CBPS design.)

bounds give close values throughout the time period of interest, and thus the exact CBPS reliability can be told reasonably precisely. The reliabilities of the original CBPS (i.e., the CBPS without redundancy) and the compatible RECBAR [2] are also provided for comparison, where their links are assumed totally fault-free (an optimistic case), and the PE in the CBPS without redundancy is assumed to have failure rate 0.8λ (because it involves four ports rather than six). While our design can tolerate up to 2^s PE failures with one at a level using 2^s total spare PE's, the RECBAR can tolerate at most s PE failures (limited to a single level) using s spare PE's. As a result, our proposed CBPS exhibits a significant reliability improvement over the original CBPS, and is more reliable than the RECBAR. The reliability gap between our design and the RECBAR becomes wider as the system size grows.

V. Layout Considerations

The conventional VLSI grid model [10] assumes that a circuit cell and an interconnection wire are comparable in size. In practice, this is often invalid since the circuit cell of a system is much larger than the interconnection wire [11]. A PE involves much more components than a link, so in the following analysis, we make use of the modified VLSI grid model where the ratio of PE width to link width, g, is assumed to be >> 1 (say, 10 or larger), and where at most two links and one PE meet at any point in the grid. The distance between any two PE's or links is one unit or more, and the links run parallel to any one of the two grid axes.

Consider the original CBPS with $L = 2^s$ levels (see Fig. 1). It is easy to verify that the number of vertical tracks needed between stage i and stage $i+1$ is 2^{i+1} (Fig. 7(a) shows an example layout of PE's and their interconnections between stages 1 and 2). Suppose that the wrap-around connection pattern is realized as depicted in Fig. 8 (note that the wrap-around connection pattern is identical in both the original CBPS and our proposed CBPS); then additional 2^s vertical tracks are needed in front of stage 0, giving rise to totally $\sum_{i=0}^{s-1} 2^{i+1} + 2^s = 3 \times 2^s - 2$ vertical tracks. Under the situation

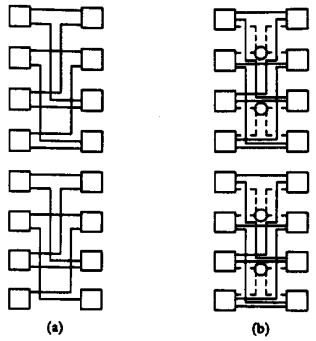

Fig. 7. An example layout pattern for PE's and their interconnections between stages 1 and 2 of (a) the original CBPS, and (b) the proposed CBPS with switches on the additional links shown.

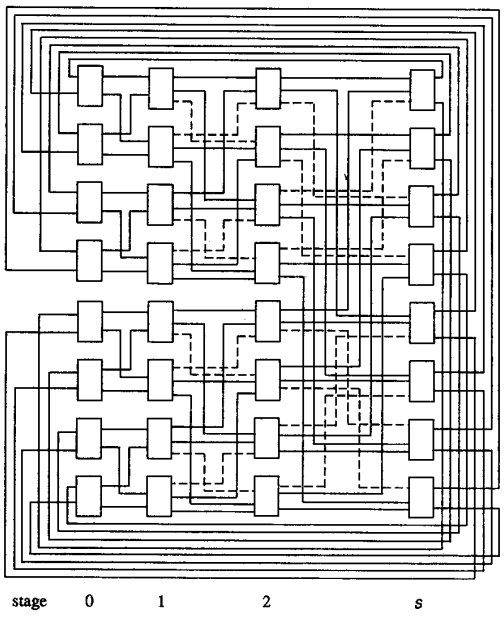

Fig. 8. An example layout of the proposed CBPS and its wrap-around connections.

of g no less than a reasonable constant (say 3 in Fig. 7(a) so that three horizontal links can be accommodated within two neighboring PE's at the same level without taking any extra vertical space), the area needed for the CBPS layout equals

$$(sg + 3 \times 2^s - 2) \times \left(g2^s + 2^{s+1}\right) \tag{3}$$

where the first term indicates the width of the layout, while the second term is the layout height (in which 2^{s+1} is the number of horizontal tracks for wrap-around connections).

Because of the specific structure of the proposed CBPS, it is possible to arrange its layout in a way that no extra vertical tracks are necessary for added links between stages i and $i+1$, for all $0 \le i < s-1$, as shown in Fig. 8. Now, if PE width is larger than the width of a control switch (a common case), then the switches on those added links take no extra vertical or horizontal space [for between two consecutive

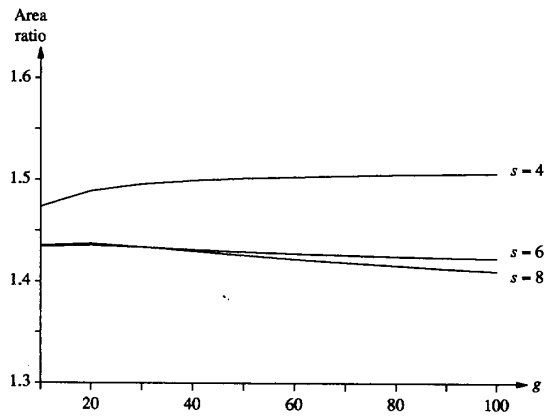

Fig. 9. Area ratio versus g for three different system sizes.

stages, at most one such switch exists in every level; see Fig. 7(b)]. The stage of spare PE's and the links between stages $s-1$ and s, however, take extra vertical spacing. An example layout of the proposed CBPS and its wrap-around connections is demonstrated in Fig. 8. Assuming that the ratio of PE width to link width become $g\prime$ [which is greater than g since the PE now has six ports and four surrounding switches, see Fig. 4(a)], then, the layout area for the proposed CBPS is equal to

$$(sg\prime + 3 \times 2^s - 2 + g\prime + 2^s) \times \left(g\prime 2^s + 2^{s+1}\right) \qquad (4)$$

where the amount $g\prime + 2^s$ in the first term is due to extra vertical spacing for spare PE's and their incoming links.

From (3) and (4), it is readily observed that, for a large s and any given constant g and $g\prime$, both layouts take an identical area of $O(2^{2s} = L^2)$. The proposed CBPS occupies the same area, in the big-O sense, as the original CBPS for a large system implemented entirely on a single chip/wafer.

It certainly is interesting to compare area space actually needed for the proposed CBPS and that for the original CBPS under various s's and g's. We calculate the area ratios of the proposed CBPS layout to the original CBPS layout for three different system sizes under $g\prime = 1.1g$. The results are depicted in Fig. 9. Notice that the results are quite conservative because a PE plus its four surrounding switches in our proposed CBPS is assumed to take 21% (i.e., $1.1^2 = 1.21$) more area than a PE in the original CBPS. It can be found from these curves that in practical situations, the proposed CBPS needs insignificant extra area as compared with the original CBPS. When the width ratio is 20, for example, the proposed CBPS with 2^4 levels occupies 49% more area than the original CBPS of a compatible size. For any given g, the area ratio decreases consistently as s grows, indicating that a larger system incurs less area overhead. On the other hand, the area ratio changes slowly for any given system when g increases. The proposed CBPS seems fairly suitable for VLSI/WSI implementation.

VI. Concluding Remarks

In this paper, we have addressed a fault-tolerant circular butterfly parallel system realized by adding a spare PE to each level and introducing extra connections and switches among PE's. A distributed reconfiguration procedure for such a design has been presented, which efficiently reconfigures itself in response to an arising fault. The design is shown to enjoy a significant improvement in its reliability. Compared with another known fault-tolerant design, the RECBAR [2], our proposed CBPS involves more spare PE's but is also more reliable. It is illustrated that the proposed design, due to its specific structure, requires only moderate extra layout area when compared with an original CBPS, under practical situations where PE width is much larger than link width.

The proposed design can tolerate at most one fault per level, since only one spare PE is added to each level. For a very large system designed for a long mission time, we can add more than one spare to every level so as to enhance its reliability even further. With this provision, a level may tolerate more than one fault. The way of placing those spares dictates the degree of reliability improvement, and also the extra area needed. For example, if a second column of spares is added to the middle of our proposed CBPS, then the resulting design is expected to exhibit high reliability improvement, but at the cost of large extra area. When the second column of spares is moved leftwards, potential reliability improvement is reduced because now in each level, a spare is shared by more PE's than the other spare, but area overhead involved also decreases since the outgoing links of the second column of spare PE's then take less area. It appears interesting to investigate an optimal design (in terms of cost-effectiveness) with multiple "columns" of spare PE's.

Appendix
Derivation of the Lower and Upper Bounds on $q_j(i_{j1})$

We observe that a newly arising fault in B_j would introduce no more than totally $2^{j+1} - 2$ new critical-E PE's, since it introduces at most 2 new critical-E PE's in stage $j-1$, 4 in stage $j-2$, etc. (see the triangles in Fig. 5). After PE <3, 5> had failed in B_3 of Fig. 5, for example, the subsequent failure can introduce from 0 (if the failure is PE <3, 1>) to 2 + 4 = 6 (if the failure is, say PE <3, 6>) new critical-E PE's. Let us examine the number of critical-E PE's in every stage of one S_{j-1}, after i_{j1} faults occurred in B_j. The number is no more than $\min(2^0 * i_{j1},\ 2^{j-1})$ in stage $j-1$, no more than $\min(2^1 * i_{j1},\ 2^{j-1})$ in stage $j-2$, etc., since there are only 2^{j-1} PE's in each stage of S_{j-1}.

In general, in the presence of i_{j1} faults in B_j, the number of critical-E PE's in stage m of one S_{j-1} is no more than $\min(2^{j-1-m} * i_{j1},\ 2^{j-1})$, or equivalently, the number of noncritical-E PE's in the stage is no less than $2^{j-1} - \min\left(2^{j-1-m} * i_{j1}, 2^{j-1}\right)$. The lower bound on $q_j(i_{j1})$ is then calculated on the basis of the following: In the presence of i_{j1} faults in B_j, the number of surely tolerable fault patterns while distributing i_{j2} (or i_{j3}) faults to one S_{j-1} (or the other S_{j-1}) of S_j equals $\sum_{m=0}^{j-1} \binom{\zeta_m}{i_{j2}}$ (or $\sum_{m=0}^{j-1} \binom{\zeta_m}{i_{j3}}$), where ζ_m is $2^{j-1} - \min\left(2^{j-1-m} * i_{j1}, 2^{j-1}\right)$ and $\binom{r}{s} = 0$ for $r < s$. This means that conservative estimation results from distributing all the i_{j2} (or i_{j3}) faults to the noncritical-E PE's in any single stage of S_{j-1}.

Let us consider again the critical-E PE's in S_j, when i_{j1} faults exist in B_j. Since two faults may introduce the same set of critical-E PE's in stage $j-1$ of S_{j-1} (such as PE <3, 5> and PE <3, 1>) but never introduce a smaller number of critical-E PE's than any single fault, the number of critical-E PE's in stage $j-1$ of S_{j-1} is no less than $\lceil i_{j1}/2 \rceil$, where $\lceil y \rceil$ is the smallest integer greater than or equal to y. From the nature of critical-E PE's, we can easily verify the following three observations: i) the number of critical-E PE's in every stage is no less due to multiple faults in B_j than due to any single fault in B_j; ii) the number of critical-E PE's is no less in a stage than in any later stage; and iii) the number of critical-E PE's in each stage m of either one of the two S_{j-1}'s is *even*, for $m \leq j-2$. The number of critical-E PE's in every stage m of S_{j-1} due to a single fault in B_j, denoted by ρ_m, is simple to estimate iteratively.

Now, from observation i), it is clear that the number of critical-E PE's in every stage m of S_{j-1} due to any multiple faults in B_j is $\geq \rho_m$. On the other hand, from observations ii) and iii), the number of critical-E PE's in stage $m-1$ of S_{j-1} can be obtained from that in stage m, for $0 < m < j$. Since a lower bound on the number of critical-E PE's in stage $j-1$ of S_{j-1} is known, we can estimate a lower bound on the number of critical-E PE's in every stage $<j-1$, stage by stage, starting with stage $j-2$. Let the number calculated this way for stage m be σ_m, then, the exact number of critical-E PE's in every stage m of S_{j-1} is $\geq \max(\rho_m, \sigma_m)$. Thus, when there are i_{j1} faults existing in B_j, the number of noncritical-E PE's in an S_{j-1} satisfies

$$\eta_j(i_{j1}) \leq N_{j-1} - \sum_{m=0}^{j-1} \max(\rho_m, \sigma_m). \tag{5}$$

The upper bound of $q_j(i_{j1})$ is obtained by making use of (2) and (5), with Ξ_{j2} and Ξ_{j3} estimated conservatively to be the numbers of distinct fault patterns each of which has at least two faulty PE's in a level. Note that this estimation on Ξ_{j2} and Ξ_{j3} is conservative because the system certainly fails if any level involves two or more faulty PE's, and may fail even when every level has no more than one faulty PE.

REFERENCES

[1] F. S. Wong and M. R. Ito, "A loop-structured switching network," *IEEE Trans. Comput.*, vol. C-33, pp. 450–455, May 1984.

[2] V. Balasubramanian and P. Banerjee, "A fault tolerant massively parallel processing architecture," *J. Parallel Distributed Comput.*, vol. 4, pp. 363–383, 1987.

[3] N. Koike and K. Ohmori, "MAN-YO: A special purpose parallel machine for logic design automation," in *Proc. 1985 Int. Conf. Parallel Processing*, Aug. 1985, pp. 583–590.

[4] L. M. Napolitano, Jr., "The design of a high performance packet-switched network," *J. Parallel Distributed Comput.*, vol. 10, pp. 103–114, Oct. 1990.

[5] C.-L. Wu and T.-Y. Feng, "On a class of multistage interconnection networks," *IEEE Trans. Comput.*, vol. C-29, pp. 694–702, Aug. 1980.

[6] G. B. Adams III and H. J. Siegel, "The extra stage cube: A fault-tolerant interconnection network for supersystems," *IEEE Trans. Comput.*, vol. C-31, pp. 443–454, May 1982.

[7] P. Banerjee, S.-Y. Kuo, and W. K. Fuchs, "Reconfigurable cube-connected cycles architectures," in *Proc. 16th Int. Symp. Fault-Tolerant Comput.*, July 1986, pp. 286–291.

[8] I. Koren, Z. Koren, and D. K. Pradhan, "Designing interconnection buses in VLSI and WSI for maximum yield and minimum delay," *IEEE J. Solid-State Circuits*, vol. 23, pp. 859–866, June 1988.

[9] P. K. Lala, *Fault Tolerant and Fault Testable Hardware Design.* Englewood Cliffs, NJ: Prentice-Hall, 1985.

[10] C. D. Thompson, "A complexity theory for VLSI," Ph.D. dissertation, Dep. Comput. Sci., Carnegie Mellon Univ., 1980.

[11] M. S. Krishnan and J. P. Hayes, "A normalized-area measure for VLSI layouts," *IEEE Trans. Comput.-Aided Design*, vol. 7, pp. 411–419, Mar. 1988.

[12] I. Koren and M. A. Breuer, "On area and yield considerations for fault-tolerant VLSI processor arrays," *IEEE Trans. Comput.*, vol. C-33, pp. 21–27, Jan. 1984.

[13] M. Blatt, "Effects of switch failure on soft-configurable WSI yield," in *Proc. 1990 Int. Conf. Wafer Scale Integration*, Jan. 1990, pp. 152–159.

[14] R. R. Koch, "Increasing the size of a network by a constant factor can increase performance by more than a constant factor," in *Proc. 29th Annu. Symp. Foundations of Comput. Sci.*, Oct. 1988, pp. 221–230.

[15] E. Upfal, "An O(log N) deterministic packet routing scheme," in *Proc. 21st Annu. ACM Symp. Theory of Comput.*, May, 1989, pp. 241–250.

[16] T. Leighton and B. Maggs, "Expanders might be practical: Fast algorithms for routing around faults on multibutterflies," in *Proc. 30th Annu. Symp. Foundations of Comput. Sci.*, Oct. 1989, pp. 384–389.

Nian-Feng Tzeng (S'85–M'86–SM'92) received the B.S. degree in computer science from National Chiao Tung University, Taiwan, the M.S. degree in electrical engineering from National Taiwan University, Taiwan, and the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign in 1978, 1980, and 1986, respectively.

He is currently an Associate Professor in the Center for Advanced Computer Studies at the University of Southwestern Louisiana, Lafayette. From 1986 to 1987, he was a member of technical staff at AT&T Bell Laboratories, Columbus, OH. His current research interest is in the areas of parallel and distributed processing, high-performance computer systems, and fault-tolerant computing.

Dr. Tzeng is a member of Tau Beta Pi, a member of the Association for Computing Machinery, and the recipient of the outstanding paper award of the 10th International Conference on Distributed Computing Systems, May 1990.