SCIREA Journal of Computer
http://www.scirea.org/journal/Computer
October 30, 2023
Volume 8, Issue 3, June 2023
https://doi.org/10.54647/computer520386

Self-restructuring Mesh-connected Processor Arrays through Spares on Moved Diagonals, Direct Replacement and Built-in Circuits

Itsuo Takanami[1], Masaru Fukushi[2,*]

[1]Department of Technology in the former times, Yamaguchi University, Ube, Japan

[2]Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube, Japan

*Corresponding author: mfukushi@yamaguchi-u.ac.jp

Abstract

We present a self-reconfiguring scheme for $N \times N$ mesh-connected processor arrays (PAs) with N spares where faulty PEs are directly replaced by spare PEs functionally located on the diagonals which may be moved. This replacement is formalized as a matching problem in graph theory. Then, the necessary and sufficient condition that all the faulty PEs in a PA are replaced (repaired) at the same time is given. Using the condition, a restructuring algorithm is given. By computer simulation, it is shown that the survival rates and the probabilities of the arrays increase so much, comparing with those of the existing network structures with the same number of spare PEs. The scheme is realized by digital circuits which can be built in a PA. The scheme may be useful in enhancing especially the run-time reliability and availability of PAs in mission critical applications where first self-reconfiguration is required without an external host computer and manual maintenance operations.

Keywords: *fault-tolerance, mesh array, direct replacement, self-restructuring, built-in circuit*

1. Introduction

Recently, high-speed and high-quality technologies for processing many kinds of information have become essential and will become more and more necessary in the future. For such needs, as VLSI technology has developed, how to realize massively parallel computing systems has been studied in the literature, e.g., [4]-[9], [22]-[26], and so on, where entire or significant parts of PEs and interconnections among them are implemented on a single chip or wafer. Therefore, the yield and/or reliability of the system may become drastically low.

In these situations, one of the most important and fundamental issues that must be addressed for such PAs is defect/fault tolerance. If a single PE fails to perform its assigned task correctly, due to some defects/faults, the entire computation will result in failure.

This manuscript concerns fault-tolerant systems consisting of many processor elements (PEs), that is, mesh-connected parallel computer systems.

A mesh-connected processor array (PA) is a kind of form of massively parallel computing systems. Mesh-connected PAs consisting of processing elements (PEs) have regular and modular structures which are very suitable for most signal and image processing algorithms.

To restore the correct computation capabilities of PAs with faults, it must be reconfigured appropriately so that the defective PEs are eliminated from the computation paths, and the working PEs maintain correct logical connectivity among them. Various strategies to reconfigure a faulty physical system into a fault-free target logical system are described in the literature, e.g., [4]-[9], [22]-[26]. Some of these techniques employ very powerful reconfiguring systems that can repair a faulty PA with almost certainty, even in the presence of clusters of multiple faults. However, the key limitation of these techniques is that they are executed in software programs to run on an external host computer and they cannot be designed and implemented efficiently within a PA chip as dedicated circuits. If a faulty PA can be reconfigured by a built-in circuit, the system down time of the PA is significantly reduced. Furthermore, the PA will become more reliable when it is used in such an environment where the fault information cannot be monitored externally through the boundary pins of the chip and manual maintenance operations are difficult.

As far as we know, the first attempts to develop self-restructuring systems (SRSs) were made by Negrini et al. [10] and Sami and Stefanelli [11]. The design approach of their repair control scheme begins with a heuristic algorithm that involves only some local reconnection operations. Mazumder et al. [12] developed an automatic SRS using an analog neural network for mesh-connected PAs with spares on one row and one column by which faulty PEs are directly replaced. Takanami et al. also developed automatic SRSs [13], [14] for mesh-connected PAs using single track switches (STSs) proposed by Kung et al. [4]. The arrays they dealt with have two rows and two columns of spares, i.e., four linear arrays of spares around a PA to cope with faults of low reliable PEs.

On the other hand, if PEs are fairly reliable, it is expected that the smaller number of spares will be sufficient for retaining the reliability of an array so high and additional control schemes as well as control circuits will become simple. From this expectation, the authors proposed SRSs for PAs with spares in one row and one column [17]–[20]. In these schemes, $2N$ spares are used for arrays with size of $N \times N$. Further, the case where the number of spares is less, that is, N, was discussed for STS structure in [21].

This paper deals with the case that the number of spares is N for an $N \times N$ array. As an arrangement of N spares for this case, a simple one is to locate them on a side of an array called singe-side spare scheme (SS scheme). Another to be presented here is to locate them on the diagonal of an array, which is called a diagonal spare scheme (DS scheme). Since how faulty PEs are replaced in SS scheme is simple and further its array reliability is less than that of DS scheme as shown in Fig. 2, in Section II, we describe how faulty PEs are replaced in DS scheme. For this scheme, two methods which are called direct replacement (DR) and STS will be considered. STS method was discussed in [21]. STS method has an advantage in that physical distances among logically adjacent PEs after reconfiguration (compensation) are bounded by a constant while those in DR method are proportional to the array sizes. On the other hand, as seen in Sect. 4, DR method realizes much higher reconfiguration probabilities than STS method. So, here we focus on DR method.

In Sect. 2, we present the fixed diagonal case of DR method and the moved diagonal one as its valiant. In Sect. 3, we formalize the strategies for deciding that by which healthy spares faulty PEs should be replaced as a matching problem in graph theory and present an algorithm for reconfiguring the arrays with faulty PEs in a convenient form to realizing built-in circuits for reconfiguration. In Sect. 4, the survival rates (successfully reconfigured rates) and array reliabilities (successfully reconfigured probabilities) as the measures to evaluate the proposed

method are shown. In Sect. 5, the built-in circuits to realize the proposed method is presented. Sect. 6 is a conclusion.

2. DS scheme and moved diagonal method

Fig. 1(a) shows SS scheme in which a faulty PE is replaced by a spare in the same row. Fig. 1(b) shows how PEs with spares are arranged in an array for DS scheme. If a PE p_{ij} located at the i-th row and j-th column is faulty, it is compensated for by either the spare S_i or S_j. For this arrangement, several compensation methods may be considered according to how an array is restructured using spares. Here, two methods are introduced. One is to directly replace a faulty PE with a spare, which is called DR method. Another is to replace a faulty PE with a spare by shifting PEs as in [4] and [9], which is called STS method.

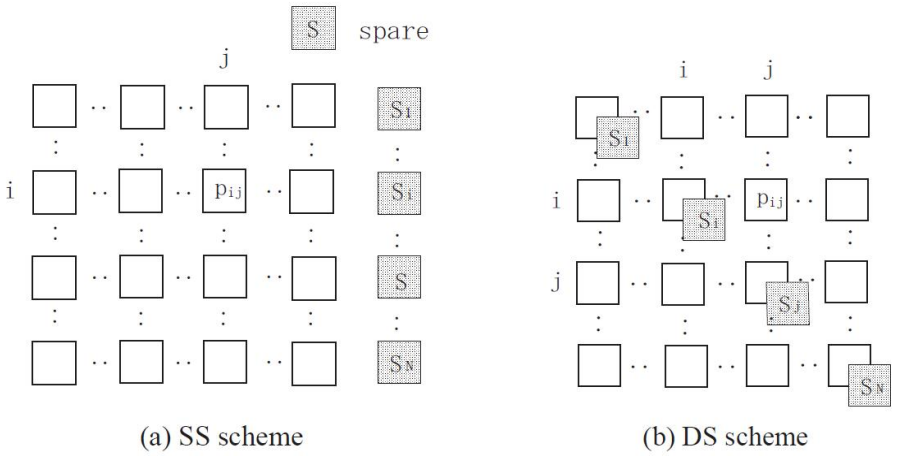


Figure 1. Arrangement of PEs and spares

STS method has an advantage in that physical distances among logically adjacent PEs after reconfiguration (compensation) are bounded by a constant while those in DR method are proportional to the array sizes. On the other hand, as seen in Sect. 4, DR method realizes much higher reconfiguration probabilities than STS method. So, here we focus on DR method. In DR method, the replacement of faulty PEs by spares on diagonal will be characterized using a bipartite graph in graph theory.

Hereafter, we give some notations.

Notation 1:

- PE(i, j) ($1 \le i \le N$) ($1 \le j \le N$) denotes the PE located at location (i, j), i.e., the i-th row and j-th column of an $N \times N$ PA.

- S_j ($1 \le j \le N$) denotes a spare on a diagonal of PA whose location is assigned in the moving process to be described later. S_j is denoted as PE(0, j) for convenience of explanation.
- f_{ij} ($0 \le i \le N$) ($1 \le j \le N$) denotes the faulty state of PE(i, j) where f_{ij} = 1 and = 0 means that PE(i, j) is faulty (healthy), respectively.
- If a faulty PE is replaced by a spare PE, it is said to be **repaired**, otherwise **unrepaired**.
- For a set of faulty PEs (including spares) which is called a fault pattern, if all the faulty PEs in the set can be repaired at the same time, the fault pattern as well as the array with the fault pattern is said to be repairable, otherwise unrepairable. Note that a faulty spare is repaired by itself.

For DR method, we formalize a strategy for deciding that faulty PEs should be replaced by which healthy spares as a matching problem in graph theory.

For the array in Fig. 1(b), we construct the following bipartite graph G called a compensation graph for a fault pattern P.

Let $V_f = \{p_{ij} \mid 0 \le i \le N, 1 \le j \le N$, PE($i$, j) is faulty$\}$, $V_s = \{p_{01}, ..., p_{0N}\}$ which is a (vertex) set of spares on the diagonal of the array. Then, $G = (V, E)$, where $V = V_f \cup V_s$, $E = \{(p_{ij}, p_{0i}), (p_{ij}, p_{0j}) \mid 0 \le i \le N, 1 \le j \le N$, PE($i$, j) is faulty$\}$. V_f is called a (vertex) set of faulty PEs and E a set of edges implying replacement relation, respectively. Note that a faulty spare PE is considered to be replaced by itself.

It is clear that the following holds.

Lemma 1 A fault pattern P is repairable by replacement if and only if a matching from V_f to V_s exists. For such a matching M, faulty PE(i, j) is replaced by spare S_j if $(p_{ij}, p_{0i}) \in M$, and by spare S_j if $(p_{ij}, p_{0j}) \in M$.

Notation 2:

Let $G = (V, E)$ be a bipartite graph where $V = V_1 \cup V_2$ and $V_1 \cap V_2 = \phi$. For S ($\subseteq V_1$), let $\psi(S) = \{v \in V_2 \mid (w, v) \in E, w \in S\}$. The degree of a vertex u which is the number of edges incident to u is denoted as $deg(u)$.

It is seen that the degree of any faulty vertex in a compensation graph is equal to or less than two. Using the fact, the repairability condition is given as follows [19].

Theorem 1 (Repairability theorem): Let $G = (V, E)$ be a bipartite graph such that $V = V_1 \cup V_2$, $V_1 \cap V_2 = \phi$ and $E \subseteq V_1 \times V_2$ where the degree of any vertex in V_1 is equal to or less than two. We partition the maximal subgraph of G with the vertex set $V_1 \cup \psi(V_1)$ into connected components and denote the vertex sets in V_1 of the connected components as C_1, C_2, ..., C_m (for each C_p, $C_p \subseteq V_1$, $\psi(C_p) \subseteq V_2$, and for $i \neq j$, $(C_i \cup \psi(C_i)) \cap (C_j \cup \psi(C_j)) = \phi$). Then, the repairability condition is as follows.

There exists a matching from V_1 to V_2 if and only if $|C_i| \leq |\psi(C_i)|$ for all C_i holds, where $|C|$ means the number of elements in C. □

Proof: see [19].

According to Theorem 1, we can judge whether a PA is repairable. Theorem 1 is also expressed in a convenient form for restructuring a PA, which is used in the algorithm MDRALG to be presented in Sect. 3.

Theorem 2: (i) For any vertex v of degree 1 in V_2 and $(w, v) \in E$, let $G' = (V', E')$ be the graph obtained by removing $\{w, v\}$ from V and the edges incident to w or v. Then, there exists a matching from V_1 to V_2 in G if and only if there exists a matching from $V_1 - \{w\}$ $(= V_1')$ to $V_2 - \{v\}$ $(= V_2')$ in G'. (ii) Let the degree of any vertex in $\psi(C_i)$ be equal to or greater than two. Then,

1. For some C_i, if there is a vertex in $\psi(C_i)$ whose degree is greater than two, there exists no matching from V_1 to V_2.

2. For some C_i, if the degree of every vertex in $\psi(C_i)$ is two and there exists a vertex of degree 1 in C_i, there exists no matching from V_1 to V_2.

3. For all C_is, if the degree of every vertex in $\psi(C_i)$ is two and there exists no vertex of degree 1 in C_i, there exists a matching from V_1 to V_2. □

Proof: see [19].

Property 1:

(i) If a nonspare faulty PE(i, j) is at the same location as that of a spare, $deg(p_{ij})$ is one and otherwise two. Further, the degree of a faulty spare is one.

(ii) For any $v \in V_1$, $deg(v)$ does not change before and after the operation (i) in Theorem 2 has been executed since edges incident to v are untouched.

3. Moved diagonal restructuring algorithm (MDR-ALG)

We present a method to logically move positions of spares for an $N \times N$ PA. For k ($1 \le k \le N$), a location to which S_j ($1 \le j \le N$) is assigned is defined by the equation below and denoted as $Loc_k(S_j)$.

$$Loc_k(S_j) = \begin{cases} (j - k+1, j) & \text{for } k \le j \\ (N - k+1+j, j) & \text{for } j < k \end{cases} \quad (1)$$

Table 1 illustrates $Loc_k(S_j)$'s for k's ($1 \le k \le 4$) and j's $1 \le j \le 4$. For k, a set of $Loc_k(S_j)$'s is called a moved diagonal with index k and denoted as DA(k).

Table 1. $Loc_k(S_j)$'s for the case of $N = 4$

k	$Loc_k(S_1)$	$Loc_k(S_2)$	$Loc_k(S_3)$	$Loc_k(S_4)$
1	(1, 1)	(2, 2)	(3, 3)	(4, 4)
2	(4, 1)	(1, 2)	(2, 3)	(3, 4)
3	(3, 1)	(4, 2)	(1, 3)	(2, 4)
4	(2, 1)	(3, 2)	(4, 3)	(1, 4)

Fig. 2(a) visually shows the moved diagonal DA(k). If the parts I and III in Fig. 2(a) are moved to the right sides of the parts II and IV, Fig. 2(b) is obtained. Then, Fig. 2(b) has the same structure as Fig. 1(b) as well as the replacing relation. So, a faulty PE is replaced by one of the spares which are located in horizontal and vertical directions of the location of the faulty PE. This is called a moved diagonal method (**MD-method**). The network structure for moving logical position of spares will be presented in Sect. 5

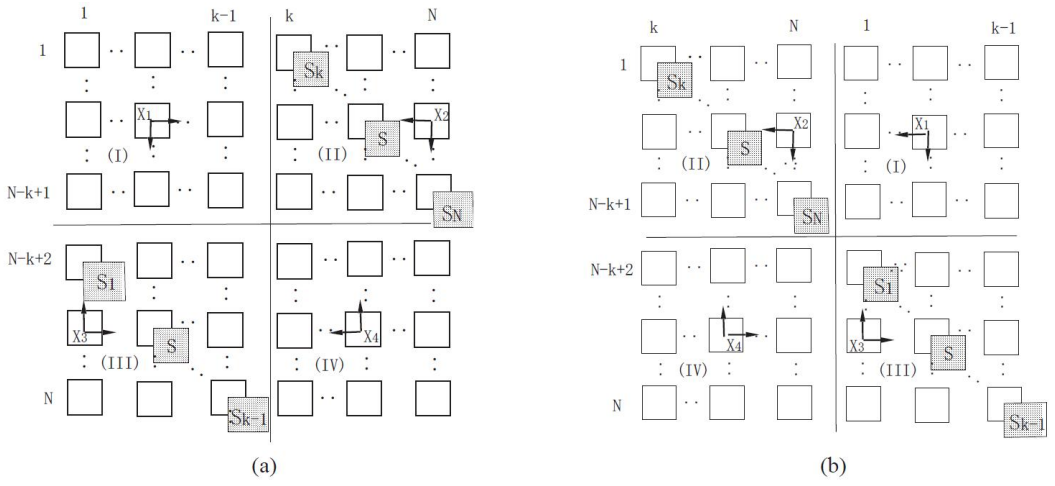


Figure 2. Arrangement of spares on the moved diagonal (DA(k))

A restructuring algorithm for MD-method to be introduced in the following is called a moved diagonal restructuring algorithm (shortly written as **MDR-ALG**) which is executed to PA with logically moved spares as in Fig. 2.

Notation 3:

- A PA with spares on DA(k) is denoted as PA(DA(k)).
- For PA(DA(k)), let $\mathrm{Loc}_k(\mathrm{S}_j) = (r, c)$. Then, the r-th row and c-th column of PA(DA(k)) are denoted as $\mathrm{R}(\mathrm{S}^k_j)$ and $\mathrm{C}(\mathrm{S}^k_j)$, respectively. For example, $\mathrm{R}(\mathrm{S}^k_1)$ is the $(N - k + 2)$-th row and $\mathrm{C}(\mathrm{S}^k_1)$ is the 1-st column in Fig. 2(a).
- The number of unrepaired faulty PEs (including S_j) through the row $\mathrm{R}(\mathrm{S}^k_j)$ and column $\mathrm{C}(\mathrm{S}^k_j)$ is denoted as $n_f(\mathrm{S}^k_j)$. For simplicity, $n_f(\mathrm{S}_j{}^k)$ are often written like $n_f(\mathrm{S}_j)$ if no confusion occurs.
- A nonspare PE at the location $\mathrm{Loc}_k(\mathrm{S}_j)$ is denoted as $\mathrm{PE}(\mathrm{Loc}_k(\mathrm{S}_j))$.

MDR-ALG

- Let the size of PA be $N \times N$.
- Let P be a fault pattern including faulty spares.

Step 1: Set k to 1.

Step 2: Construct the compensation graph $G = (V, E)$ for PA(DA(k)) with P.

Step 3: Let $V_1' = V_f$, $V_2' = V_s$ and $E' = E$. Let $M = \phi$ (empty set).

Step 4: While there is a vertex v with $deg(v) = 1$ in V_2', do this step.

For $(w, v) \in E'$, let $M = M \cup \{(w, v)\}$, $\widehat{E} = \{(w, \hat{v}) \mid (w, \hat{v}) \text{ in } \mathrm{E}'\}$, $E' = E' - \widehat{E}$, $V_1' = V_1' - \{w\}$ and $V_2' = V_2' - \{v\}$ (i.e., match w with v, and delete all the edges incident to w together with w and v).

Step 5: If $V_1' = \phi$, M is a matching from V_1 to V_2. Then, go to Step 8.

Step 6: If there is a vertex v in V_2' whose degree is more than 2 or there is a vertex v in V_1' with $deg(v) = 1$ and $k < N$, increase k by 1 and go to Step 2. Otherwise, it is judged that there is no matching and go to Step 8.

Step 7: Let $\widehat{G}$ be the compensation graph obtained. Then, there is a matching in $\widehat{G}$ and find a closed cycle in each derived connected component $\widehat{C}_i$, from which just two different matching

in $\widehat{C}_i$ are derived. Choose one of them which is denoted as M_i. Let $M = M \cup \{\cup_i M_i\}$. Then M is a matching from V_f to V_s.

Step8: The algorithm ends.

The following is the detailed process for executing MDR-ALG by hardware, where spare PEs are physically located on the upper side as shown in Fig. 5.

(Execution of MDR-ALG (shortly written as EMDR-ALG) for an $N \times N$ PA)

Step I: Set k to 1.

Step II: Set a diagonal to DA(k).

Step III: While there is an unrepaired faulty PE through R(S_j) and C(S_j) with $n_f(S_j) = 1$ for some S_j, do (i) below.

(i) For each spare S_j, get $n_f(S_j)$. This is done in parallel for all the spares S_j's. Then, replace (repair) a faulty PE through R(S_j) and C(S_j) with $n_f(S_j) = 1$ by the spare S_j and set $n_f(S_j)$ to 0.

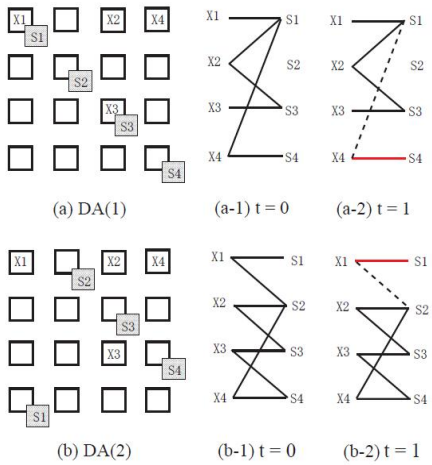
This step corresponds to Step 4 in MDR-ALG.

Step IV: For some S_j, if $n_f(S_j) \geq 3$ or (for $n_f(S_j) = 2$, S_j or the PE($Loc_k(S_j)$ is faulty and unrepaired) and $k < N$, increase k by 1 and go to Step II. Otherwise, the PA with faults is unrepairable and go to Step VI.

This step corresponds to Step 6 in MDR-ALG.

Step V: The array is repairable. Then, proceed to find closed cycles and determine directions of replacement for the unrepaired faulty PEs in the cycles.

Step VI: The algorithm ends.



	DA(1) t		DA(2) t	
	0	1	0	1
$n_f(S1)$	③	②	1	0
$n_f(S2)$	0	0	3	2
$n_f(S3)$	②	②	2	2
$n_f(S4)$	1	0	2	2
	unrepairable		repairable	

(c) The values of $n_{f(Sj)}$'s

Figure 3. Flows of compensation graphs together with $n_f(S_j)$'s for PA(DA(k)) (k=1 and 2)

Fig. 3 shows the flow of the compensation graphs for a fault pattern P = {X1, X2, X3, X4} in execution of MDR-ALG where the red lines denote the matched pairs and t the number of times that (i) in Step III has been executed. Fig. **3**(c) shows the flow of $n_f(S_j)$'s ($1 \le i \le 4$) where $n_f(S_j)$'s with circle mark imply that S_j's or PE($Loc_k(S_j)$)s are faulty, from which it can be judged that the pattern can't be repaired by the spares in DA(1) (i.e., k = 1)) but DA(2) (i.e., k = 2).

This suggests that MD-method may increase the survival rates and array reliabilities.

4. Evaluation

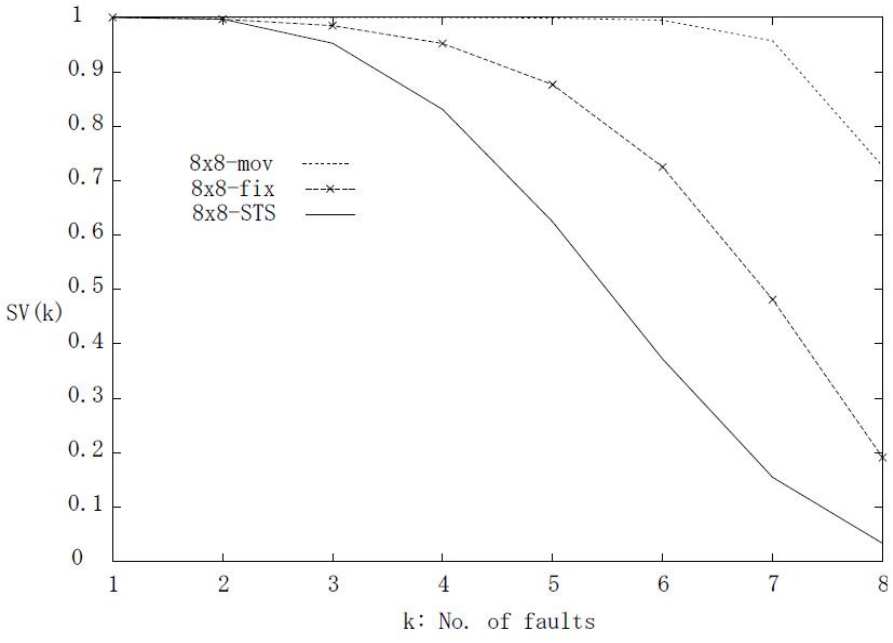
In this section, to estimate the performance of MDR-ALG, we have executed Monte Carlo simulations using a PC with Borland C++ Compiler 5.5. Here, it is assumed that all the PEs including spares may become uniformly faulty. Then, 10^6 random fault patterns each with k faulty PEs for $1 \le k \le N_s$ are generated provided that each PE in an array has the same reliability p, where N_s is the number of spare PEs. Then, we have evaluated survival rates (SVs) and array reliabilities (ARs), comparing with those of the existing algorithms for arrays with the same number of spares. Here, SV (k) is the ratio of the number of repairable fault patterns each with k faulty PEs to the number of fault patterns each with k faulty PEs examined, which is the probability estimated by simulation that fault patterns each with k faulty PEs are repaired. The array reliability $AR(p)$ is defined as the sum of all the probabilities each of which is computed as the product of probability that a fault pattern is repaired and one that the pattern occurs under the condition that each PE may be healthy with equal probability p, as below. Then, SV (k) and $AR(p)$ are given as

$$SV(k) = \frac{N_{rep}(k)}{N_{pat}(k)},$$

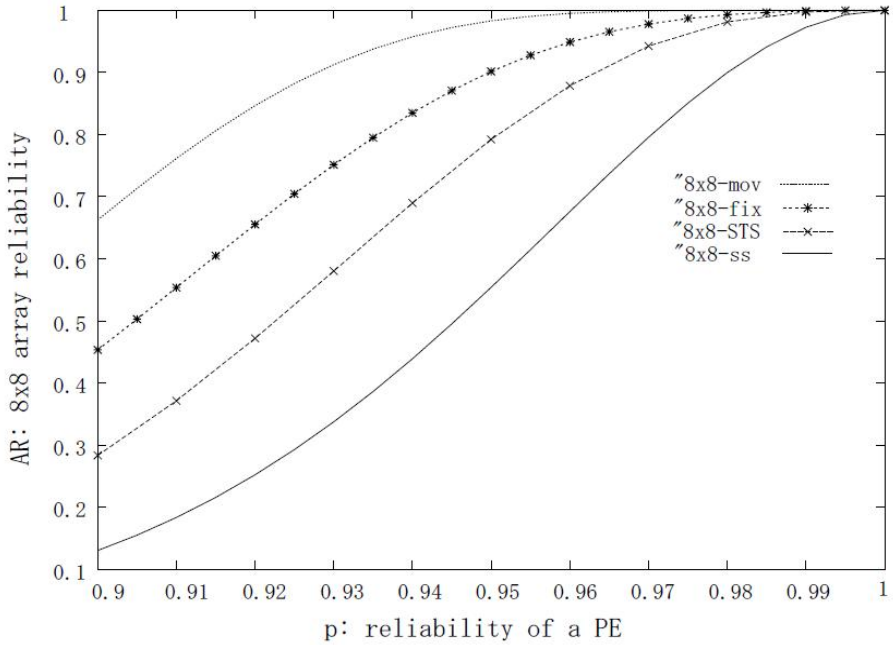
$$AR(k) = \sum_{k=0}^{N_s} {}_{N_a}C_k \cdot SV(k) \cdot p^{Na-k} \cdot (1-p)^k,$$

where $N_{rep}(k)$, $N_{pat}(k)$, and N_a are the number of fault patterns which have k faulty PEs and are judged to be repairable, the number of examined fault patterns which have k faulty PEs, and the number of all PEs, respectively. Note that $p^{Na-k} \cdot (1-p)^k$ is the probability that a fault pattern with k faulty PEs occurs, ${}_{N_a}C_k = \frac{N_a!}{(N_a-p)!\,k!}$ the number of fault patterns each with k faulty PEs.

Fig. 4 shows SVs and ARs of PAs with size of 8 × 8, where '-mov' denotes one for MDR-ALG, '-fix' for MDR-ALG with diagonal fixed to DA(1), -STS for STS method and '-ss' for ss-scheme, respectively. It is seen that the ARs and SVs for MDR-ALG (denoted as 8x8-mov) are much higher than those for STS as well as the MDR-ALG with DA(1), i.e., for PA(DA(1)).



(a) Survival rate



(b) Array reliability

Figure 4. Survival rates and array reliabilities for 8 x 8 arrays

5. Hardware realization

Fig. 5 is an illustration of the network structure for PA(DA(1)) in which the spares are physically located on the upper side and the switches around each PE. To see a scene that faults are replaced by spares, it is shown that the PE with mark X1 is replaced by the spare S1 through the red lines. A PE together with the switches surrounded with the dotted square line is called a PE module (shortly written as **PE-M**) which is depicted as in Fig. 6. Then, the structure in Fig. 5 is concisely depicted as in Fig. 8. The switches in a PE-M consist of H-, V- and D-switches (denoted as H-, V- and D-sw's). D-sw's are ON if the PE-M is logically located on the diagonal, otherwise OFF. The states of the switches (H- and V-sw's) denoted by circular marks around a PE are determined and depicted as shown in Fig. 7, depending on whether the PE in the PE-M is healthy or faulty and the directions in which it is replaced by a spare.

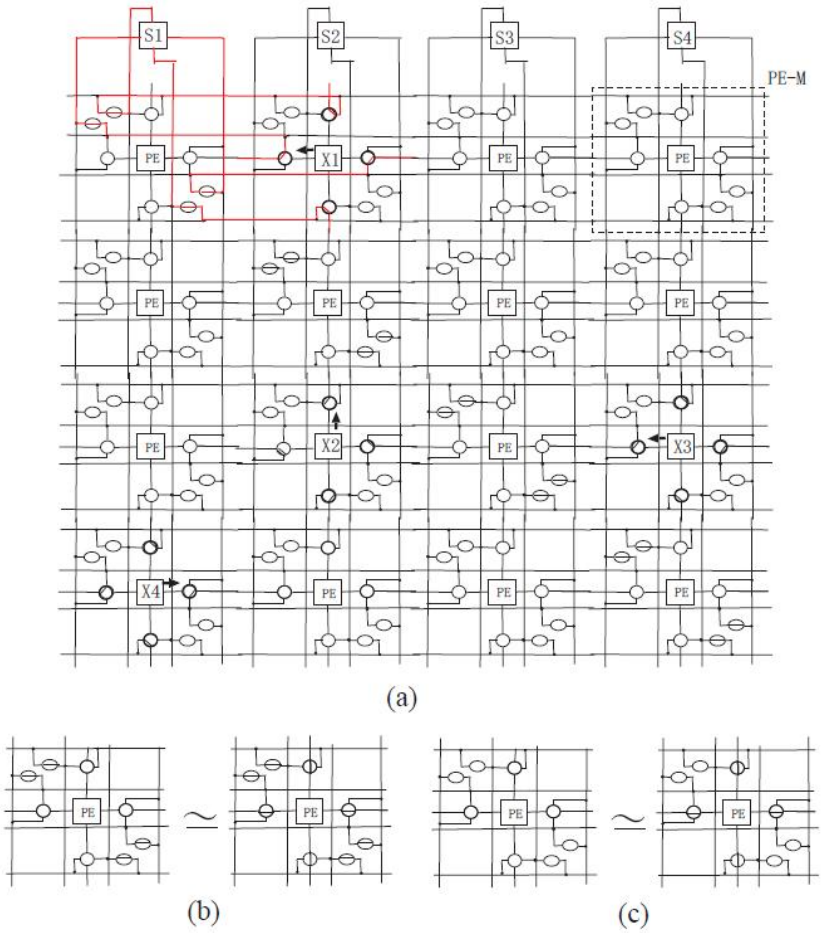


Figure 5. (a) Network structure with spares on DA(1) where PEs with (without) x marks are faulty(healthy) where in (b) and (c), those in the left side are the simplified expressions of those in the right side

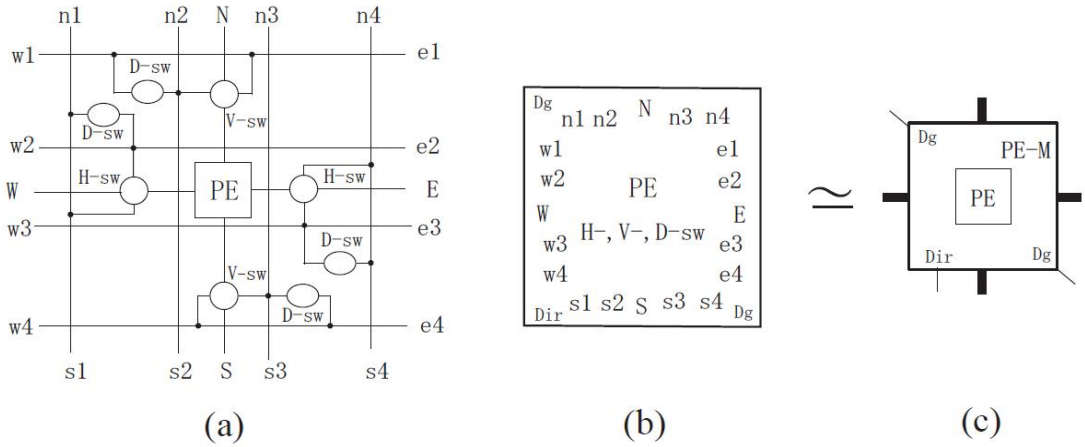


Figure 6. PE Module (PE-M)

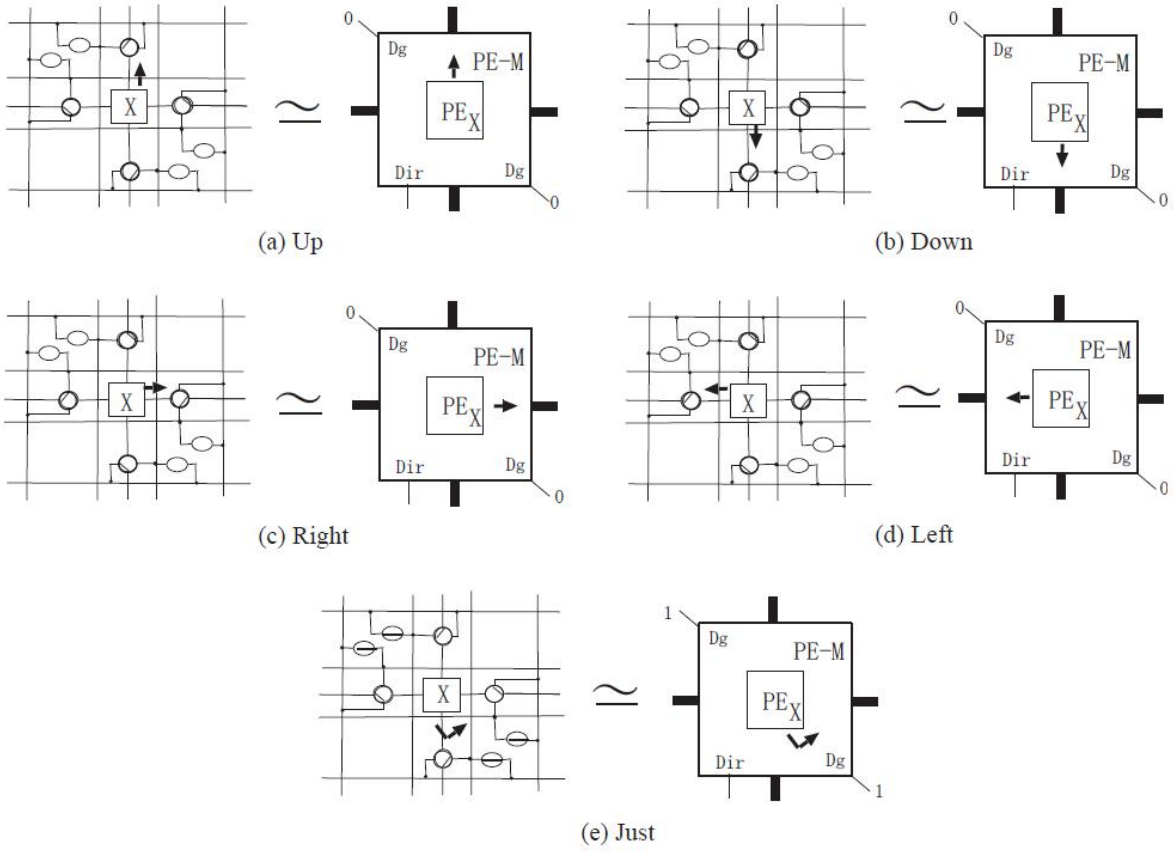


Figure 7. The states of switches with circular marks around a PE according to the directions in which it is replaced

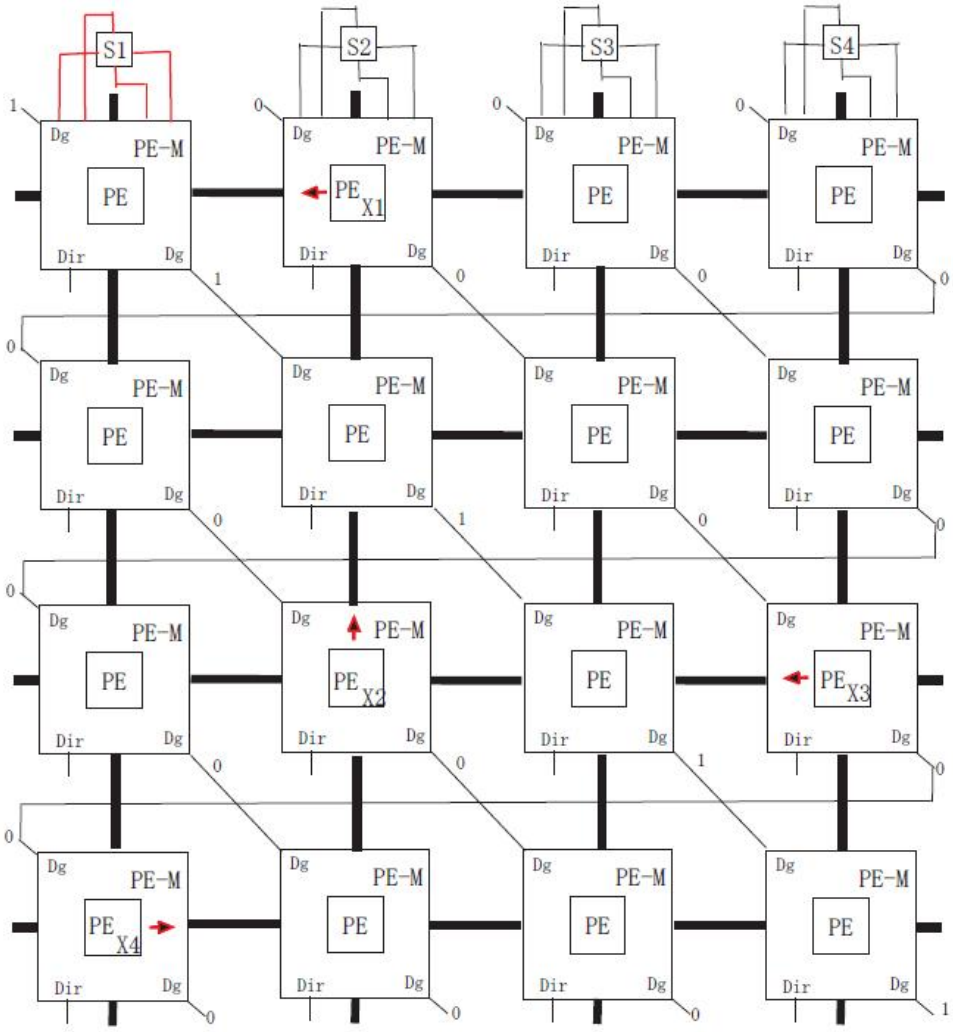


Figure 8. The concise expression of the network structure by PE-Ms

Then, it is seen that the following property holds. This property is effectively used in designing NET-1 to be described.

Property 2:

(i) The states of the H- and V-sw's for UP, DOWN and JUST directions are the same, where JUST means that the logical address of a PE is the same as that of a spare, which is called 'UDJ-sw state'.

(ii) The states of the H- and V-sw's for Right and Left directions are the same, which is called 'LR-sw state'.

MDR-ALG will be realized by digital circuits for the network structure shown in Fig. 5. This is done by two circuits. One is NET-1 and another is NET-2. First, NET-1 outputs a signal whether a PA with faulty PEs is repairable, i.e., all the faulty PEs are replaced by healthy spare PEs at the same time. If the array is not repairable, '1' is output from the terminal

UNRP, otherwise "0" is output. When UNRP = 0, NET-2 determines a direction toward a spare by which each unrepaired faulty PE in the closed cycles obtained in Step V of EMDR-ALG should be replaced.

The following is the more detailed outline of hardware realization for an $N \times N$ PA.

(Detailed outline of hardware realization)

- Let P be a fault pattern for an $N \times N$ PA in which S_i's ($1 \leq i \leq N$) are spare PEs physically located on the upper side.

Step I: Set k to 1.

Step II: Set UNREC to 1.

Step III: While ($k \leq N$ and UNREC = 1), for PA(DA(k)) do from Steps IV to VI.

Step IV: Set UNREC to 0.

Step V: For PA(DA(k)), get $n_f(S_j)$ ($1 \leq j \leq N$). This is done in parallel for all the j's. If $n_f(S_i) = n_f(S_j) = 1$ ($i \neq j$), (i) single faulty PE is in R(S_i) and C(S_j), or (ii) in C(S_i) and R(S_j). Then, we make it be replaced by S_j for the case (i) and S_i for the case (ii), and set $n_f(S_i)$ and $n_f(S_j)$ to 0s. Note that if a spare S_k is faulty, it is replaced by itself. This step corresponds to Step III in EMDR-ALG.

Step VI: For some S_j, if (a) $n_f(S_j) \geq 3$ or (b) for $n_f(S_j) = 2$, S_j or the PE($Loc_k(S_j)$) is faulty and unrepaired, increase k by 1 and set UNREC to 1 and go to Step III. This step corresponds to Step IV in EMDR-ALG.

Step VII: If UNREC = 1, the array with the faults is unrepairable and the process is ended. Otherwise, repairable and proceed to the next step.

Step VIII: Each unrepaired faulty PE in closed cycles described in Step 7 in MDR-ALG is replaced as follows. This step corresponds to Step V in EMDR-ALG.

To begin with the leftmost column, do the following.

(i) Check whether there are unrepaired faulty PEs in the column. This is done by sending a signal "1" from the lowest row in the column toward the upper. If it is confirmed that there is none, go to the column next to the right. Otherwise, there are unrepaired one or two faulty PEs but not more than two in the column. Let denote the start of the cycle by A and the end by B. Then, six location patterns of A and B as in Fig. 9 are possible where the oblique lines show PEs on diagonal. In designing a circuit, we make the location pattern (d) in Fig. 9 not

occur. The PE A which receives the signal “1” from the lower sends signal “1”s to the left and right and is replaced by a spare PE logically located on the diagonal.

- A faulty PE which has received a signal “1” from the upper or lower sends signal “1”s to the right and left. Then, the sw’s around the PE are set to UDJ-sw state, and it is replaced by a proper spare PE on the diagonal.
- A faulty PE which has received a signal “1” from the left or right sends signal “1”s to the upper and lower. Then, the sw’s around the PE are set to LR-sw state, and it is replaced by a proper spare PE on the diagonal.
- A healthy or repaired PE only passes the signal which it has received.

(ii) Finally, a signal “1” will reach the PE B. If the column checked is the rightmost column, this process is ended. Otherwise, go to the column next to the right and go to (i).

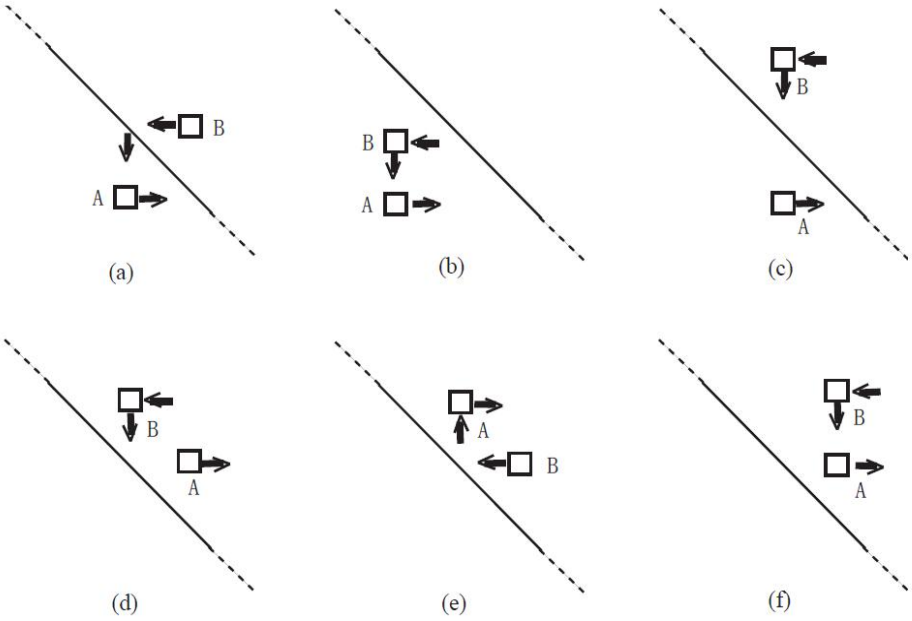
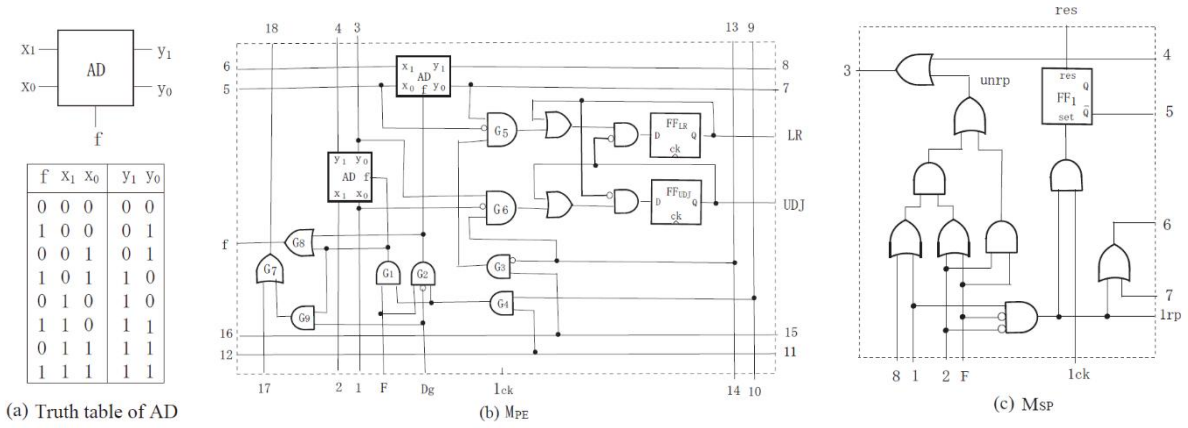


Figure 9. Possible location patterns of A and B in a cycle where arrows are implied directions of replacement

First, we show a digital circuit NET-1 which realizes from Steps I to VII above. Next, we show a digital circuit NET-2 which decides the directions of replacements for faulty PEs in closed cycles in Step VIII above.

Fig. 10 shows NET-1 which consists of modules MPE, MSP and Dg-sw where MPE is shown in (b) and MSP in (c). The fault signal of a nonspare PE (spare PE) is input to the terminal F of MPE (MSP) as shown in Fig. 10.



f	X_1	X_0	y_1	y_0
0	0	0	0	0
1	0	0	0	1
0	0	1	0	1
1	0	1	1	0
0	1	0	1	0
1	1	0	1	1
0	1	1	1	1
1	1	1	1	1

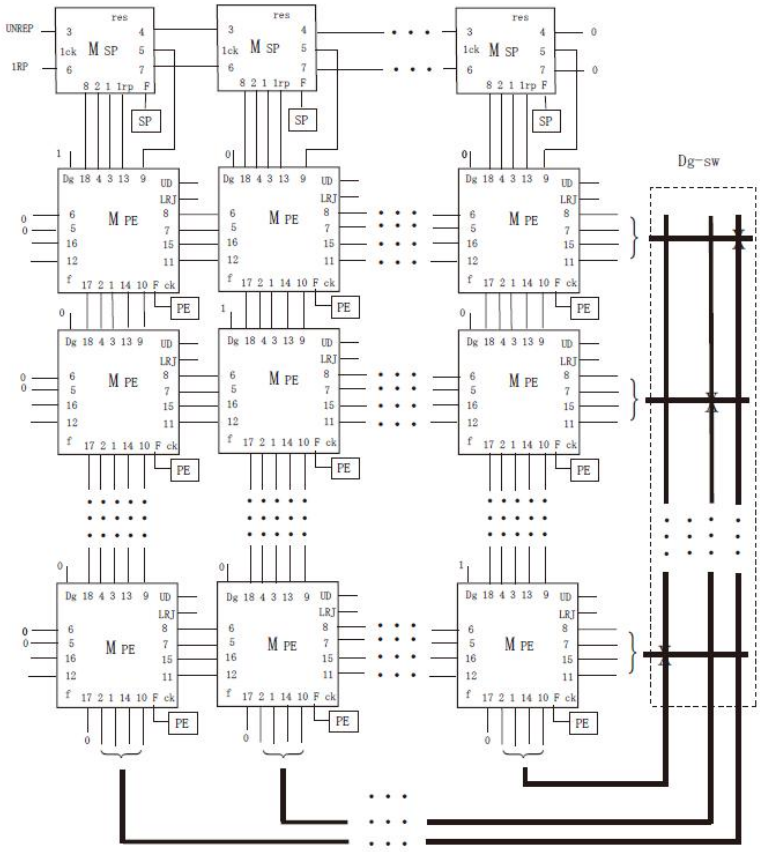


Figure 10. NET-1

Notation 4:

- PE(*x,y*) (including a spare PE) denotes the PE in the *x*-th row and *y*-th column, where PE(0,*y*) denotes the spare PE on the upper side of an array.

- Input signal to a terminal number n is denoted as i_n. The output signal from a terminal number m is denoted as o_m. These symbols are also used as the terminal names if no confusion occurs.

- $i(x,y)$ and $o(x,y)$ denote the input and output to/from MPE(x,y).

We explain the functions of the modules and Dg-sw in the following.

- The function of AD

AD is used to count the number of faulty PEs through R(S_i) and C(S_i) for each spare S_i to check whether $n_f(S_i) > 2$. To do so, AD adds binary numbers $(x_1x_0)_2$ and $(0f)_2$, and outputs a binary number $(y_1y_0)_2$ but the sum is upper-limited to 3, that is, $(11)_2$ if it is greater than 2, as shown in Fig. 10(a).

- The function of MPE

1. When $n_f(S_i) = 1$, the fault state of PE(a, b) = $\overline{x_0} \cdot y_0$ for (a, b) in RC(S_i) where x_0 and y_0 are the signal values of the terminals of C_1 in $M_{PE(a, b)}$.

2. If the signals $i_9 = i_{11} = 1$ (so initially as will be described in 1 of the behavior of NET-1), $o_f = i_F$, otherwise $o_f = 0$.

3. If $D_g = 0$, i.e., PE is not on the diagonal, $(o_8o_7)_2 = (i_6i_5)_2 + (0f)_2$ and $(o_4o_3)_2 = (i_1i_2)_2 + (0f)_2$. If $D_g = 1$, i.e., PE is on the diagonal, $(o_8o_7)_2 = (i_6i_5)_2$ and $(o_4o_3)_2 = (i_1i_2)_2 + (0f)_2$.

4. LR and UDJ are defined as in Table 2. Then, we have LR = $\overline{i_{13}} \cdot i_{15}$ and UDJ = i_{13}. The data of either of LR or UDJ are held correspondingly when either of them is 1.

Table 2. Truth table for LR and UDJ

f	i_{13}	i_{15}	LR	UDJ
1	0	0	0	0
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1
0	*	*	0	0

- The function of MSP

1. $i_8(0, y) = 1$ if and only if PE(S_i) with $D_g = 1$ is faulty where S_i = PE(0, y).

2. When FF_1 is rest, $o_5 = 1$. When 1rp = 1 and 1ck is fed, the output Q of FF_1 becomes 1 and hence, $\overline{Q}$ becomes 0. Table 3 is the truth table for unrp and 1rp which is obtained from Steps V and VI in the detailed outline. From Table 3, it is seen that 1rp = 1 if and only if i_F = 0 and $(i_2 i_1)_2 = 1$, and unrp = 1 if and only if (a) $(i_2 i_1)_2 \geq 3$ or (b) $i_F = 1$ and $(i_2 i_1)_2 = 1$ or (c) i_8 = 1 and $(i_2 i_1)_2 = 2$. (In Table 3, the subscripts (a), (b) and (c) are attached to the corresponding rows like (iv)(a), (vi)(b), (xi)(c)) and so on.

Table 3. Truth table for unrp and 1rp (∗ 's are 'don't care')

	$i8$	F	$i2$	$i1$	unrp	1rp
(i)	0	0	0	0	0	0
(ii)	0	0	0	1	0	1
(iii)	0	0	1	0	0	0
(iv)(a)	0	0	1	1	1	0
(v)	0	1	0	0	0	0
(vi)(b)	0	1	0	1	1	0
(vii)(a)	0	1	1	0	1	0
(viii)(a)	0	1	1	1	1	0
(ix)	1	0	0	0	*	*
(x)	1	0	0	1	0	1
(xi)(c)	1	0	1	0	1	0
(xii)(a)	1	0	1	1	1	0
(xiii)	1	1	0	0	*	*
(xiv)(a)	1	1	0	1	1	0
(xv)(a)	1	1	1	0	1	0
(xvi(a))	1	1	1	1	1	0

From Table 3, the logical equations of unrp and 1rp are give by

unrp = $(F + i_2)(i_8 + i_1) + F \cdot i_2$,

1rp = $\overline{F} \cdot \overline{i_2} \cdot i_1$.

- The function of Dg-sw

Fig. 11 illustrates the state of Dg-sw to determine the positions of DA(∗) for a 4 × 4 PA. Through the Dg-sw, the terminals 8, 7, 15 and 11 of the MPE's in the rightmost column are connected to the terminals 2, 1, 14 and 10 in the MPE's in the lowest row, respectively, according to the positions at which the Dg signals are given to the MPE's at the top row.

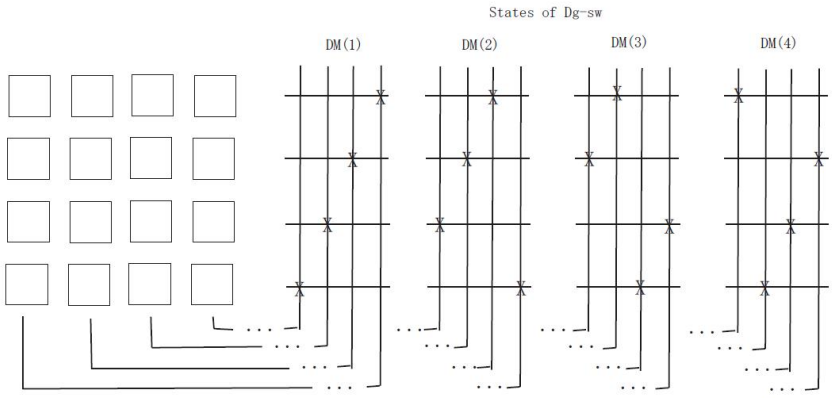


Figure 11. States of Dg-sw according to DA(i)s ($1 \leq i \leq 4$)

(The behavior of the circuit NET-1)

1. Initially, all the flip-flop FFs in MPE's and the shift-registers are reset, i.e., o_5 of each MSP is 1. Then, i_9 of each MPE is 1, and $(i_2 i_1)_2$ of each MSP shows n_f(SP) excluding the fault state of SP, though it is upper-limited to 3.

2. The behavior of NET-1 is controlled by the clocks input to the terminals 1ck's of MSP's.

3. If 1RP = 1, there is an MSP (= MPE(0, y)) whose output $o_{1\mathrm{rp}}$ is 1. Let the logical address of SP be (x, y). Then, the $o_{1\mathrm{rp}}$ is fed to $i_{13}(1, y)$, which is output from $o_{14}(N, y)$ which is input to $i_{15}(x, N)$ through Dg-sw. The signal 1 passes y-th column (i.e., C(SP)) and x-th row (i.e., R(SP)). Then, there is a single faulty PE whose physical address is (a, y) or (x, b). Let it denote as (a', b'). Then, the output of either G_5 or G_6 in MPE(a', b') becomes 1. In this situation, when a clock through 1ck is input to NET-1, the FF_1 of the MSP is set to 1 and $o_5(0, y)$ becomes 0, which is input to the terminal i_9 of MPE(1, y) which passes y-th column (C(SP))and x-th row (R(SP)) through Dg-sw. Then, a faulty PE(a', b') is replaced by the SP where the direction of replacement is determined by the output of either G_5 or G_6 (see Table 2).

4. If UNREP = 0, unrp's of all MSP's are 0's, which indicates that the array with the faults is repairable. Next, if there is S_k such that $n_f(S_k) = 2$, go to the process to find closed cycles (even if there may not be such cycles) together with the directions of replacing unrepaired faulty PEs in the cycles. This process corresponds to Step VIII in the detailed outline and is executed by NET-2 shown in Fig. 15.

NET-2 is a network consisting of C_{PE}'s, a shift-register, flip-flops and gates as shown in Fig. 15, and decides the directions of replacing unrepaired faults with spares in executing Step VIII in the detailed outline.

- The function of C_{PE}

C_{PE} has three subcircuits whose functions are switched according to the cases of (F = 1 and Dg = 0), (F = 0 and Dg = 0) and Dg = 1.

1. The terminal *f* of MPE in NET-1 is connected to the terminal F of C_{PE}.

2. If i_F (= o_f) is 0, the internal structure becomes as shown in Fig.12(b), i.e., the signals pass through.

3. If i_F = 1 and Dg = 0, the internal structure becomes as shown in Fig. 12(a). Then,

(i) (a) the signal through x_3 from the top or (b) x_1 or v_1 from the bottom are transferred to the left through y_4 and the right through y_2, and the signal is stored in the flip-flop FFLR which indicates that the direction of replacement is to the left or right. This scene is seen in Fig. 13(i).

(ii) (a) The signal through x_4 from the left or (b) x_2 from the right are transferred to the lower through y_1 and the upper through y_3, and the signal is stored in the flip-flop FFUDJ which indicates that the direction of replacement is the top or down. This scene is seen in Fig. 13(ii).

4. If Dg = 1, the internal structure becomes as shown in Fig. 12(c). Then,

(i) If v_1 = 0, the signal through x_i (i = 1, 2, 3 or 4) is transferred to three y_j's ($j \neq i$), where the case of i = 1 is shown in Fig. 14(iii).

(ii) If v_1 = 1, the signal through x_1 is transferred only to y_3 as shown in Fig. 14(iv). The signals through x_2, x_3 and x_4 are transferred only to y_1 (see Fig. 12(c) in which the circles in the cross points of the vertical and horizontal lines mean that the logical ORed signals from the horizontal directions are transferred to the vertical directions, e.g., $y_1 = x_2 + x_3 + x_4$).

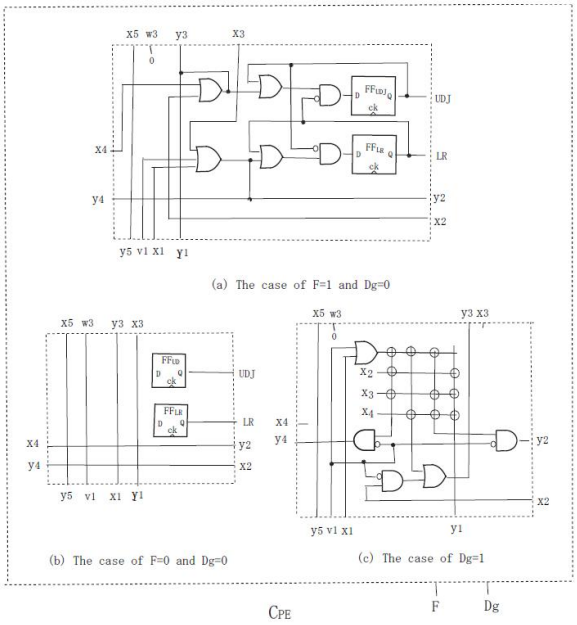


Figure 12. Inner-structure of module C_{PE}

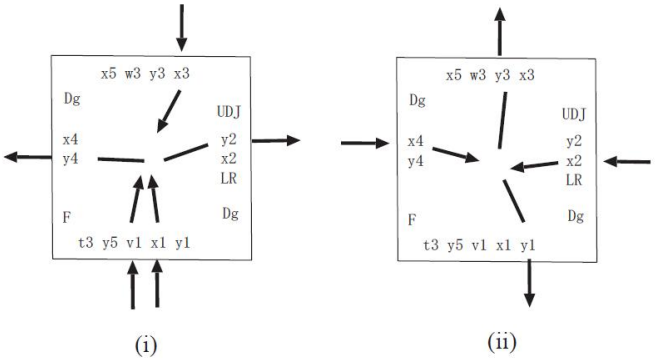


Figure 13. Signal flow of inputs and outputs in the case of F = 1 and Dg = 0

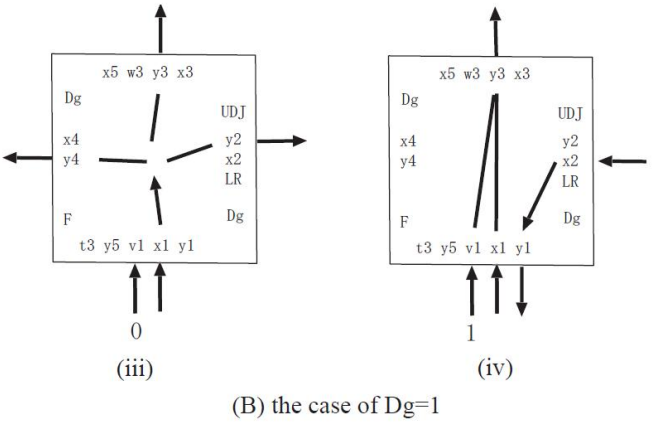


Figure 14. Signal flow of inputs and outputs in the case of Dg = 1

(The behavior of the circuit NET-2)

NET-2 operates when a given fault pattern is repairable, i.e., UNREP = 0. Note that the internal structure of C_{PE} becomes as shown in Fig. 12(b) if a PE is healthy or has been repaired in NET-1 and as shown in Fig. 12(a) if it has not yet been repaired and is in a closed cycle. Further, note that there are one or two unrepaired faulty PEs in a row or column in a closed cycle.

1. Initially, all the flip-flops are reset.

2. Signal '1' in the shift-register is shifted from the left to the right at the time when the output of G_1 is 0 and a clock pulse is fed to CLK-1.

3. Increasing *i* from 1 to *N*, the following is performed.

(i) A clock is fed to all the flip-flops in Fig. 15 through CLK-2 except ones in the shift-register.

(ii) If Qi of the shift-register becomes 1, this signal '1' is input to v1 of CPE in the bottom row of i-th column. At the time, the output of the gate G1 becomes 1 and hence a clock to CLK-1 is inhibited to be supplied to the shift register. While a clock to CLK-1 is not supplied, the signal '1' input to the i-th column behaves as follow.

- If there is no unrepaired faulty PE in the *i*-th column, the signal '1' passes through all the C_{PE}s in the column, turns back to the C_{PE} in the top row of the column (note that the terminals x_5 and y_3 are connected) and reaches the terminal y_5 of the C_{PE} in the bottom row of the column.

- If there are unrepaired faulty PEs, there are unrepaired one or two faulty PEs but not more than two in the column. Let denote the start of the cycle by A and the end by B. Then, six location patterns of A and B are possible as shown in Fig. 9. As mentioned before, the location pattern (d) in Fig. 9 does not occur. The signal '1' is fed to A and propagates in a closed cycle as mentioned in Step VIII in the detailed outline of hardware realization, finally reaches B and the terminal y_5 or y_1 of the C_{PE} in the bottom row of the column.

- The signal '1' which reaches y_1 or y_5 of the C_{PE} in the bottom row as above is memorized in the D-FF by a clock to CLK-2 and fed to the gate G_2. Then, the outputs of G_2 and G_1 become 0s, and hence, a clock to CLK-1 can pass through the gate G_{ck}.

The above will be seen in Fig. 15 as an instance for a 4×4 PA(DA(3)) where the arrows show the flow of the signal '1'.

From the explanation so far, it is seen that NET-1 and NET-2 exactly execute each step in EMDR-ALG.

5. Conclusion

It has been shown that system lifetime extremely increases by introducing the moved diagonal method. Further, it has been shown that the proposed approach can be realized by the digital network consisting of comparatively simple digital modules which can easily be embedded in a target PA to recover from the faults without the aid of a host computer for reconfiguration. This implies that the proposed method is so useful in enhancing especially the run-time reliabilities and availabilities of PAs in mission critical systems where first self-reconfiguration is required without an external host computer or manual maintenance operations.

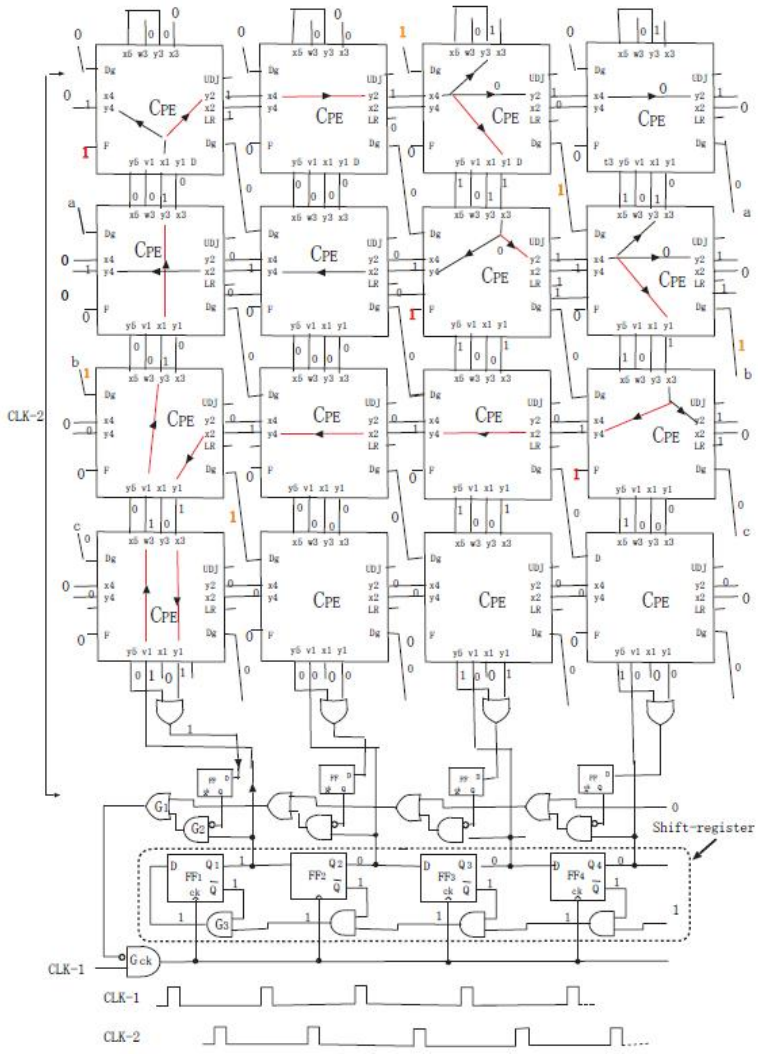


Figure 15. NET-2 for deciding the directions of replacement while executing Step VII

Reference

[1] L.W. Schaper, "Design of Multichip Modules," Proc. IEEE, Vol. 80, Issue 12, pp. 1955-1964, 1992.

[2] K. Okamoto, "Importance of Wafer Bonding for The Future Hype-Miniaturized CMOS Devices," ECS Transactions, Vol. 16, No.8, pp. 15-29, 2008.

[3] W.J. Dally and B. Towles, "Route Packets, Not Wires: On-chip Interconnection Networks," Proc. of the 38th Design Automation Conference, pp. 684-689, 2001, March, 2001.

[4] S.Y. Kung, S.N. Jean and C.W. Chang, "Fault-Tolerant Array Processors Using Single-Track Switches," IEEE Trans. Comput., Vol.38, No.4, pp. 501-514, Jan., 1989.

[5] E. Mangir and A. Avizienis, "Fault-Tolerant Design for VLSI: Effect of Interconnection Requirements on Yield Improvement of VLSI Designs," IEEE Trans. Comput., Vol.c-31, No.7, pp.609-615, July, 1982.

[6] R. Negrini, M.G. Sami and R. Stefanelli, "Fault-tolerance through reconfiguration of VLSI and WSI arrays," MIT Press series in computer systems, MIT Press, 1989.

[7] V.P. Roychowdhury, J. Bruck and T. Kailath, "Efficient Algorithms for Reconstruction in VLSI/WSI Array," IEEE Trans. Comput., Vol.39, No.4, pp.480-489, April, 1989.

[8] I. Koren and A.D. Singh, "Fault Tolerance in VLSI Circuits," IEEE Computer, Vol. 23, No. 7, pp.73-83, July, 1990.

[9] T.A. Varvarigou, V.P. Roychowdhury and T. Kailath, "Reconfiguring processor arrays using multiple-track models: the 3-tracks-1-spare-approach," IEEE Trans. Comput., Vol.42, No.11, pp. 1281–1293, Nov, 1993.

[10] R. Negrinii, M. Sami and R. Stefanelli, "Fault Tolerance Techniques for Array Structures Used in Supercomputing," IEEE Computer, Vol.19, No.2, pp.78-87, Feb, 1986.

[11] M. Sami and R. Stefanelli, "Reconfigurable Architectures for VLSI Processing Arrays," Proc. IEEE, pp.712-722, May, 1986.

[12] P. Mazumder and Y.S. Jih, "Restructuring of square processor arrays by Built-in Self-Repair Circuit," IEEE Trans. Comput. Aided Des., Vol.12, No.9, pp.1255-1265, Sep, 1993.

[13] I. Takanami, K. Kurata and T. Watanabe, "A Neural Algorithm for Reconstructing Mesh-Connected Processor Arrays Using Single-Track Switches," Proc. of Int'l Conf. on WSI, pp. 101-110, Jan, 1995.

[14] T. Horita and I. Takanami, "An FPGA Implementation of a Self-Reconfigurable System for the $1\frac{1}{2}$ Track-Switch 2-D Mesh Array with PE Faults," IEICE Trans. Inf. & Syst., Vol.E83-D, No.8, pp.1701-1705, Aug, 2000.

[15] S.Y. Lin, W.C. Shen, C.C. Hsu and A.Y. Wu, "Fault-tolerant router with built-in self-test/self-diagnosis and fault-isolation circuits for 2D-mesh based chip multiprocessor systems," Int. Jour. of Electrical Engineering., Vol. 16, No. 3, pp. 213-222, 2009.

[16] J. H. Collet, P. Zajac, M. Psarakis and D. Gizopoulos, "Chip Self-Organization and Fault-Tolerance in Massively Defective Multicore Arrays", IEEE Trans. on Dependable and Secure Computing, Vol. 8, No. 2, pp. 207-217, March, 2011.

[17] I. Takanami, "Self-Reconfiguring of $1\frac{1}{2}$-Track-Switch Mesh Arrays with Spares on One Row and One Column by Simple Built-in Circuit," IEICE Trans. Inf. & Syst., Vol.E87-D, No.10, pp.2318-2328", Oct, 2004.

[18] I. Takanami and T. Horita, "A Built-in Circuit for Self-Repairing Mesh-Connected Processor Arrays by Direct Spare Replacement," Proc. of IEEE 18th Pacific Rim International Symposium on Dependable Computing, pp. 96-104, Nov, 2012.

[19] I. Takanami, T. Horita, M. Akiba, M. Terauchi and T. Kanno, "A Built-in Self-repair Circuit for Restructuring Mesh-Connected Processor Arrays by Direct Spare Replacement," Trans. on Computational Science XXVII, LNCS 9570, pp. 97-119, April, 2016.

[20] I. Takanami and M. Fukushi, "Self-restructuring of Mesh-Connected Processor Arrays with Spares Assigned on Rotated Orthogonal Sides," Trans. on Computational Science XXXVIII, LNCS 12620, pp. 36-53, 2021.

[21] I. Takanami and M. Fukushi, "A Built-in Circuit for Self-restructuring Mesh-Connected Processor Arrays with Spares on Diagonal," Trans. on Computational Science XXXIV, LNCS 11820, pp. 109-135, 2019

[22] J. Wu, L. Zhu, P. He and G. Jiang, "Reconfigurations for Processor Arrays with Faulty Switches and Links," Proc. of 15th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing, pp. 141-148, 2015.

[23] J. Qian, Z. Zhou, T. Gu, L. Zhao and L. Chang, "Optimal Reconfiguration of High-Performance VLSI Subarrays with Network Flow," IEEE Trans. On Parallel and Distributed Systems, pp. 3575-3587, Dec., 2016.

[24] J. Qian, F. Mo, H. Ding, Z. Zhou, L. Zhao and Z. Zai, "An improved algorithm for accelerating reconfiguration of VLSI array," Integration, Vol. 79, pp. 124-132, 2021.

[25] H. Ding, J. Qian, B. Huang, L. Zhao and Z. Zai, "Flexible scheme for reconfiguring 2D mesh-connected VLSI subarrays under row and column rerouting," Journal of Parallel and Distributed Computing, Vol. 151, pp. 1-12, 2021.

[26] H. Ding, J. Qian, L. Zao and Z. Zhai, "A high-performance VSLI array reconfiguration scheme based on network flow under row and column rerouting," Journal of Parallel and Distributed Computing, Vol. 158, pp. 176-185, 2021.