There are two distinct types of faults that may occur in arrays. The first type of fault is caused by the issue of bit-flip tolerances in regular arrays, which arise when the number of filled memory cells in the array does not match the expected number. The second type of fault is caused by the issue of bit-flip tolerances in arrays with associative memory, which can cause errors in the memory and lead to incorrect results.
In the recent past, the proliferation of full-chip design has made it necessary to decompose the design into smaller, more manageable components. This is achieved through the use of design closure techniques that enable the design to be broken down into smaller, more manageable components.

One of the main problems that arise in design closure is the accurate determination of the number of PEs (processing elements) required to achieve a given level of performance. In practice, designers often underestimate the number of PEs required, leading to over-designed and inefficient systems.

Another approach to design closure is to use PEs that are capable of performing multiple tasks simultaneously. This allows for a more efficient use of resources and can lead to significant improvements in overall performance.

Consequently, the use of distributed arithmetic units can be very beneficial. However, the implementation of distributed arithmetic units in practical designs can be complex and may require significant design effort.

In addition, the use of distributed arithmetic units in these designs is the subject of ongoing research. The performance of these units can vary significantly depending on the specific implementation, and researchers are continually working to improve their efficiency and effectiveness.
and no PE has two neighbors belonging to the same group. These two

The partition is such that (i) every PE is surrounded by PE of other groups,

as opposed to (ii) the number of PE of other groups.

and (iii) PE have a ring of neighbors.

After this partitioning, there are several phases of learning where

The procedure first partitions all the PE's into seven disjoint learning groups.

We propose a distributed learning procedure in which every PE reads all its

decisions by the adjacent PE's.

2. A Distributed Learning Procedure

This concept first appeared in [14], where it was applied to combinations of
not physically the same PE or, better, the PE receives them through some CEs.
and PE are accessible in every direction. It is possible, however, that some of the neighbors
using the same linkage as close to the PE as possible. If PE the PE of the learning group is
commands with which it is possible, however, that some of the neighbors
be even more effective than the PE of the learning group is

The knowledge is achieved in two basic steps: the forming stage and the

Fault Tolerance in Hexagonal Arrays

The fault-tolerance scheme for hexagonal arrays is based on the

and square arrays [2], 3, 8].

and square arrays and provide fault-tolerance in the hexagonal

and square arrays for various applications compared to simpler topologies like linear

more suitable for various applications compared to simpler topologies like linear

more suitable for hexagonal arrays. These arrays were shown to be more robust and

section of the PE does not provide the

so that only a subset of the PE does not provide the

and section of the PE does not provide the

The conclusion of the fault-tolerance scheme in [14] is that the hexagonal array

overall yield is increased.

CORRADO ET AL. RESTRUCTURING HEXAGONAL ARRAYS

25
Figure 1. An example of partitioning the Pe and seven nearest neighbors.

Assume \( (1+1) \mod 7 \) to Pe in direction 3.
Assume \( (2+2) \) mod 7 to Pe in direction 2.
Assume \( (2+2) \mod 7 \) to Pe in direction 1.

Directions 1, 2, and 3 as follows:

After being assigned a group number, assign a group number to the neighbors in the group number 0 to the left-upper corner Pe in the whole array. Every other Pe, group number.

The resulting procedure for this example is illustrated essentially by assigning the group number.
only one type of CE, 0-3, and 1-4, respectively. Note that if we disregard absolute directions, there is
characteristic then, thus, CEs of type 1, 3, and 5 link the opposite pairs 2-5.

Figure 2. A shufflable element for the shuffle

The number of shuffling directions follows the suggested in [2], and is shown in Figure
initial & messages away from the root and six rays of CEs are thus formed.

In Figure 5 we see an example of a single family P. All of its neighbors

reached from the root P, are the members of the opposite direction.

The same direction a straight line from the root P, doesn't have any neighbors.

When a PE receives a request for a PE, it becomes a CEs of type (p - 1)

in the opposite direction and becomes a CE of type (p). This is true

for any CEs in the opposite direction. When a PE receives a request for a PE, it becomes a CEs of type (p + 1) in the opposite direction.

Assume that the PE sends a request to the PE in the opposite direction.

A single path is shown in Figures 6 and 7.

Figure 6 shows the two types of connection diagrams: (a) family P, (b) family P.

Figure 7 shows the three types of connection diagrams: (a) family P, (b) family P, (c) family P.
Prevailing them from propagating further

The above sequence of events causes the external messages produced by three

handshaking any incoming messages.

failing links the repeated action (ensuring external knowledge and becomes a CE) before

Recall that all external is initially external, and we also assume that the recon-

Figure 4: CE’s formed by a regular connection between A and B.
2.4 Multiple Faults

![Diagram of multiple faults in a circuit]

When a PE detects more than one fault, it transmits C messages to all of its neighbors.
Figure 6. The effect of multiple failures A and the connection between C and D are shown in Figure 6. The resulting configuration of CES is shown in Figure 6. When one (or more) of the neighboring CES is failed, we get a situation similar to Figure 6.
REFERENCES

Utilizing the Hexagonal Array

The family of communication links (transmission or the various algorithms) following the introduction of a scheme for fault-tolerance in hexagonal arrays has been suggested. This main

3. CONCLUSION

becomes faulty, the logical effect is the same as several simultaneous link failures.

The results of the simulations for both sequential and simultaneous multiple faults with a single PE, the resulting configuration consists of 6 × 8 HCA only.

The resulting configuration consists of 6 × 9 HCA, while in Figure 5, the resulting configuration consists of 6 × 8 HCA. While in Figure 4, the sample outputs from the simultaneous program. Notice that in Figures 4–6 are sample outputs from the simultaneous program. Notice that in

Theorem: Assume a single and multiple faults have been written.

Conclusion: Assuming an initial u \times n HCA, if PE consists of d connections each PEs become

Appendix.

Proof: See Appendix.

HCA

\[(2 - u)(2 - u)\]

\[\text{if the fault is in a PE, the resulting configuration contains all (u)}\]

\[\text{if the fault is in a link, the resulting configuration contains all (u)}\]

Proposition: Assume the array is restructured after a single fault. Then

\[\text{every PE in row i and column j is linked to two (u)}\]

We prove the correctness of the scheme for single faults and for multiple faults.

2.5 Correctness of the Scheme

JOURNAL OF VISS AND COMPUTER SYSTEMS
The result of applying Algorithm 2 to the VLSI system is:

$$\begin{align*}
C \in \{ \overline{f}, \overline{f} \} & \quad (1-f, 1-f) \\
B \in \{ \overline{f}, \overline{f} \} & \quad (f, 1-f) \\
A \in \{ \overline{f}, \overline{f} \} & \quad (1, 1)
\end{align*}$$

HCA (shown above the PE in Figure 1) is first assigned to P(1,1) in the original physical map, where P(1,1) is the target area. We assume that a new index is associated with each index in the original physical map. We call the corresponding pair of coordinates P(1,1) and C(1,1). Let us consider the following example:

- If A is the minimum physical point, then we follow the direction of the output. If the output is the minimum physical point, then we follow the direction of A.
- If we follow the minimum physical point, then we follow the positive direction.
- We assume that the physical point is the target area.

Once we find the target area of each output, we can use the step by step algorithm to find the minimum physical point.

(1) The step by step algorithm is applied to each of the steps.
(2) The step by step algorithm is applied to each of the steps.
(3) The step by step algorithm is applied to each of the steps.
(4) The step by step algorithm is applied to each of the steps.

The proof is a straightforward examination of a number of sample cases. Part (a) of the proposition is proved. Part (b) will be shown in a forthcoming paper. Part (c) will be shown in a forthcoming paper.
The new indices satisfy the following conditions. They start at 1, and so they run up to \( n - 1 \); all required.

Other connections between A, B, and C are also as required for HCA. Consider a P.E. P on A's rightmost column. P's new indices of this P.E. P are \((1 + f', 1)\) and conversely at \((1 + f' + 1 + f'')\). The new indices of C's leftmost column, \((1 + f' + 1 + f'')\), conversely at \((1 + f' + 1 + f'')\).

Groups (across the P.E. A are as required.

It is clear that within each of the 3 groups A, B, and C, all links remain as.

Figure 2: An intergroup-connected array with P.E. indices before and after tracing.
REFERENCES

1. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

2. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

3. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

4. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

5. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

6. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

7. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

8. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

9. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

10. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

11. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

12. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

13. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

14. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

15. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

16. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

17. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

18. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

19. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

20. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

21. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

22. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

23. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

24. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

25. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

26. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

27. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

28. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

29. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.

30. A. K. Jain and B. Handcock. "On the Use of the HCA by at most 1, so we remain with an HCA of size 1.2 (n-2)/n at least.