Modelling and algorithms for spare allocation in reconfigurable VLSI

S.-Y. Kuo
W.K. Fuchs

Indexing terms: Very large scale integration, Modelling, Algorithms

Abstract: One approach to enhancing the yield or reliability of large area VLSI structures has been by means of spare interconnect, logic and computational units. Although extensive literature exists concerning design for inclusion of spares and restructuring mechanisms in memories and processor arrays, little research has been published on general models and algorithms for broad classes of spare allocation and reconfiguration problems. The main contribution of the paper is that a novel bipartite graph theoretic approach to spare allocation is presented for structures with dedicated spares and direct replacement reconfiguration. Spare allocation for classes of reconfigurable structures is shown to be equivalent to either a graph matching or a graph dominating set problem. The complexity of optimal spare allocation for each of the problem classes is described, and reconfiguration algorithms are provided. Several detailed examples are presented, and implementation of the algorithms is discussed.

1 Introduction

Design for reliability and yield enhancement in large area VLSI chips and wafer scale integration, by means of spare interconnect, logic and computational units and appropriate restructuring mechanisms, has been studied extensively [1]. Until recently, there have not been extensive research and development concerning efficient general models and algorithms for allocating spare units in the presence of faults [2–7].

In this paper, it is shown that spare allocation can be modelled as a simple bipartite graph problem for many reconfigurable structures with dedicated spare units. Spare allocation for broad classes of structures is shown to be equivalent to either a graph matching or a graph covering problem. The complexity of optimal spare allocation for the problem classes is described, and spare allocation algorithms are provided.

A reconfigurable structure (e.g. a chip or wafer) with dedicated spares can be viewed as an undirected graph where nodes (vertices) in the graph represent the basic replaceable units, which can be as simple as memory cells or as complex as processors, and edges in the graph represent interconnection links between replaceable units. Each node (replaceable unit) and each edge (interconnection link) can be in one of three states: active, faulty or spare.

Two design approaches that have been used for reconfiguring around faulty nodes using spare units are direct replacement and shifted replacement [8]. In direct replacement, each faulty node is replaced by a spare node without requiring further replacements. In shifted replacement, each faulty node is replaced by one of its nonfaulty adjacent nodes, and this replacing node is again replaced by one of its adjacent nodes, and so on, until a spare node is incorporated into the structure [9]. The bipartite graph models and spare allocation algorithms described in this paper are applicable to direct replacement reconfigurable designs with dedicated spare units. Designs utilising modified forms of shifted replacement reconfiguration may not be accurately modelled by the bipartite graphs described.

A target graph (architecture) G_t is a graph representing the desired system structure with all its nodes and edges in active states. A host graph (architecture) G_h is a graph representing an interconection structure in which each node and edge can be active, faulty or spare, and which contains a subgraph isomorphic to a target graph G_t. Reconfiguration is the process of modifying the mapping of the target graph onto the host graph to compensate for the presence of faults within the host graph. An optimal reconfiguration occurs when dedicated spares are allocated for all faulty units. Reconfiguration cost is a function of the number of spares used, interconnection, system performance degradation in the reconfigured system, the time to find a reconfiguration solution, and the time spent in reconfiguring the chip or wafer. The problem addressed concerns efficient allocation of spares in these reconfigurable structures in the presence of multiple faulty units.

2 Graph theoretic spare allocation models

Many reconfigurable designs for VLSI structures, utilising dedicated spare units with direct replacement, can be classified into the following categories:

(*a*) *Single replacement:* only one faulty unit is replaced by each spare allocated.

The authors wish to express their thanks to Ken Kubiak for his assistance in developing the software implementing the spare allocation algorithm. This research was supported by the Semiconductor Research Corporation under Contract 89-DP-109.

Paper 8791E (C1), first received 18th September 1991 and in revised form 17th February 1992

S.-Y. Kuo is with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, Republic of China

W.K. Fuchs is with the Center for Reliable and High-Performance Computing, Coordinated Science Laboratory, 1101 West Springfield Avenue, University of Illinois, Urbana, IL 61801, USA

(i) Single option (SRSO): only one particular spare unit can replace the faulty unit.

(ii) Multiple option (SRMO): any one of several spare units can replace the faulty unit.

(*b*) *Block replacement:* a block of units is replaced by each spare allocated.

(i) Single option (BRSO): the faulty unit is replaceable by one type of block.

(ii) Multiple options (BRMO): the faulty unit is replaceable by several types of blocks.

The basic bipartite graph models for the above strategies are different, as discussed below.

2.1 Single replacement

2.1.1 Single option (SRSO)

In this class, each spare unit replaces one faulty unit, and each faulty unit has only one choice to select a spare unit. A host graph can be partitioned into blocks of units, with each block including a spare unit as well as spare links, such that any faulty unit in this block can be replaced by this particular spare unit only. Fig. 1 shows an example

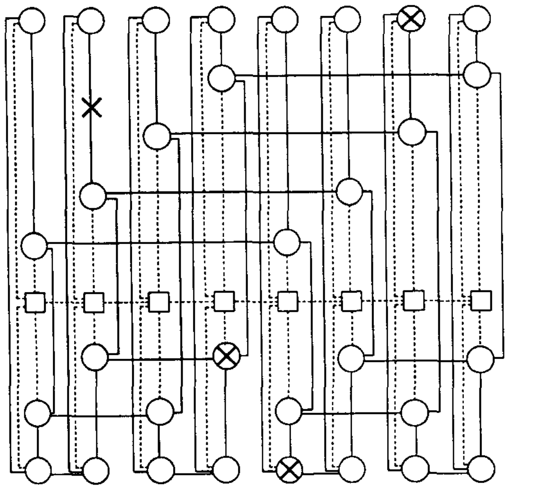

Fig. 1 *Example SRSO scheme with three faulty processors and one faulty interconnect (crossed)*

SRSO scheme developed by the authors for reconfigurable cube-connected cycle architectures [10]. Each column is a cycle of four processors represented by circles, with one spare processor, represented by a box. The normal links are represented by solid lines. The spare processor has spare links, represented by dashed lines, to every other processor unit in this cycle. Also shown are three faulty processors and one faulty interconnect, represented by crosses.

The relationship between faulty and spare units can be represented by a bipartite graph model. A bipartite graph is a graph whose node set can be partitioned into two sets, A and B, where each edge has one node in A and one node in B. It is denoted by $BG = (A, B, E)$, where E is the set of edges. Singh observed that reconfiguration can be modelled in some cases as a bipartite graph problem [11]. Let nodes in A represent faulty units and nodes in B represent spare units. An edge exists from a node α in A to a node β in B, if the faulty unit represented by α can be replaced by the spare unit represented by β and the reconfiguration path from α to β in G_h contains no faulty link. The degree of each node in A is at most one, and the degree of each node in B can be greater than one in the SRSO model. Fig. 2*a* represents the scenario in Fig. 1, with one spare unit left unconnected to indicate some spare units are not utilised. If there is a one-to-one relationship between faults and spares, as in Fig. 2*a*, the system can be reconfigured. Otherwise, the system is not reconfigurable, as in Fig. 2*b*, in which two nodes in A are connected to the same node in B.

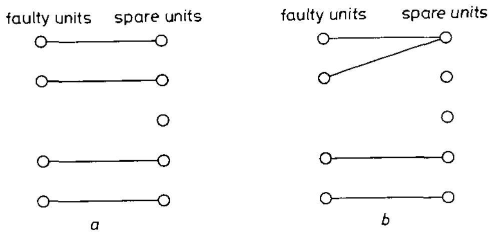

Fig. 2 *Example bipartite graph models of spare allocation for SRSO*
a Reconfigurable *b* nonreconfigurable

2.1.2 Multiple options (SRMO)

In this case, each spare unit still replaces one faulty unit, but there is more than one choice of a spare to replace a fault. An example is the reconfigurable mesh with three faulty units and one faulty link, as shown in Fig. 3 [12]. The circles in Fig. 3 represent spare units, and the squares are the units in the target graph. Each faulty unit, as indicated by ●, has either one or two choices to be replaced by a spare unit. The faulty link is indicated by ×.

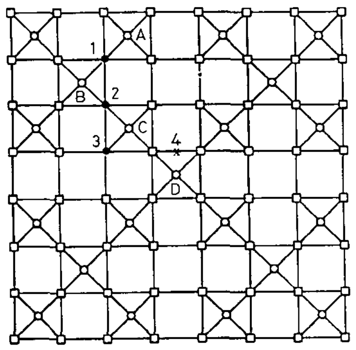

Fig. 3 *SRMO scheme with three faulty units (●) and one faulty interconnection (×)*

Fig. 4*a* shows the bipartite graph model for the example in Fig. 3. This graph is a many-to-many relationship. The degree of any node in BG can be greater than one. The spare allocation problem now becomes a matching problem. A matching M of a graph

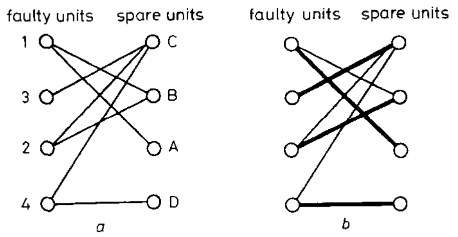

Fig. 4 *Bipartite graph model of the faulty SRMO structure in Fig. 3*

$G = (V, E)$ is a subset of the edges E, with the property that no two edges of M share the same node in V. The set of bold edges in Fig. 4*b* shows a matching for Fig. 4*a*. The number of spare units must be greater than, or equal to, the number of faulty units, otherwise the system is not reconfigurable. A matching must be found such that every faulty unit is assigned a spare. This bipartite graph matching problem is known to be solvable in polynomial time [13].

2.2 Block replacement

2.2.1 Single option (BRSO)

In block replacement, instead of replacing a single faulty unit by a single spare, a block of units (faulty and nonfaulty) is replaced by a block of spare units. Each faulty unit belongs to only one type of block which means there is only one option to select a spare to cover this faulty unit. An example is shown in Fig. 5, which is an array of units with three spare columns of units represented by three vertical lines. Also shown in Fig. 5 are six faulty units represented by circles. In this case, a block is a column of the array.

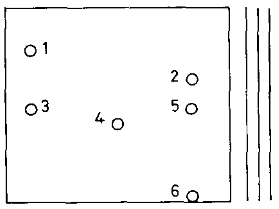

Fig. 5 *An example BRSO scheme*

An alternative type of bipartite graph, $BG = (A, B, E)$, is used here to describe this problem. Nodes in A still represent all the faulty replaceable units, but nodes in B now represent different replaceable blocks. An edge exists between two nodes α and β, if the faulty unit denoted by α in A is covered by the block denoted by β in B, and the reconfiguration path in G_h from β to the available spare block contains no faulty or unavailable link. The corresponding graph model for Fig. 5 is shown in Fig. 6. The degree of each node in A is at most one, as there is only one type of block in the BRSO class. The spare allocation is simple, in that nodes in B are assigned spare blocks until all the faulty units are replaced or no spares are left.

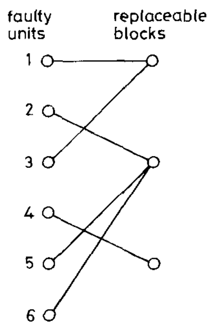

Fig. 6 *Bipartite graph model of the faulty BRSO structure in Fig. 5*

2.2.2 Multiple options (BRMO)

In this class, more than one type of block exists, and each faulty unit can belong to several sets of different types of block. There are multiple options to select different types of spare block to cover a faulty unit. The allocation of a spare block can cover more than one faulty unit. Hierarchically reconfigurable architectures typically fall into this category [12]. An example BRMO scheme, with nine faulty cells, represented by circles, is shown in Fig. 7.

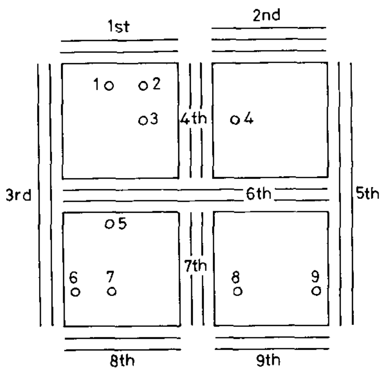

Fig. 7 *An example BRMO scheme*

Each rectangle represents an array of cells. For every faulty cell, there are several options to select either a spare row or a column to replace the row or column that contains the faulty cell. Note that there are nine different kinds of spare rows and columns represented by horizontal and vertical lines, respectively. A bipartite graph, $BG = (A, B, E)$, is used to represent the spare allocation problem. Nodes in A denote the faulty replaceable units in the system, and nodes in B denote blocks of different types. An edge exists between two nodes α and β, if the faulty unit denoted by α in A is in the block denoted by β in B, and the reconfiguration path in G_h from β to the available spare block contains no faulty link. The nodes in B are divided into sets, with each set containing a single type of block, i.e. the nodes in the same set are replaced by the same spare blocks. The bipartite model of BRMO, with nine types of blocks corresponding to the nine different kinds of spare rows or columns, for the example in Fig. 7 is shown in Fig. 8. In this model, for multiple types of block, spare allocation becomes a restricted dominating set problem with constraints [14].

Definition: A *dominating set* of a graph $G = (V, E)$ with $|V|$ vertices and $|E|$ edges in a subset $V' < V$ such that for all $u \in V - V'$ there is a $v \in V'$ for which $\{u, v\} \in E$.

In the model, all the faulty units are placed on one side of the bipartite graph, and only the blocks that contain these faulty units are placed on the other side of the graph. The objective is to replace some of the blocks with spare blocks, so that all the faulty units are replaced with spare units. Hence, the problem can be stated as finding a set of nodes in B that covers all the nodes in A, subject to the constraints that the number of nodes chosen in each block cannot exceed the number of spares available in each block.

2.3 Generalised models including costs

In the previous bipartite models, all the edges or nodes were unweighted. However, some spares may be preferred over others due to considerations involving the cost of reconfiguration, switch limitations, routing contentions and performance. Cost can be assigned to each edge or node during construction of bipartite models.

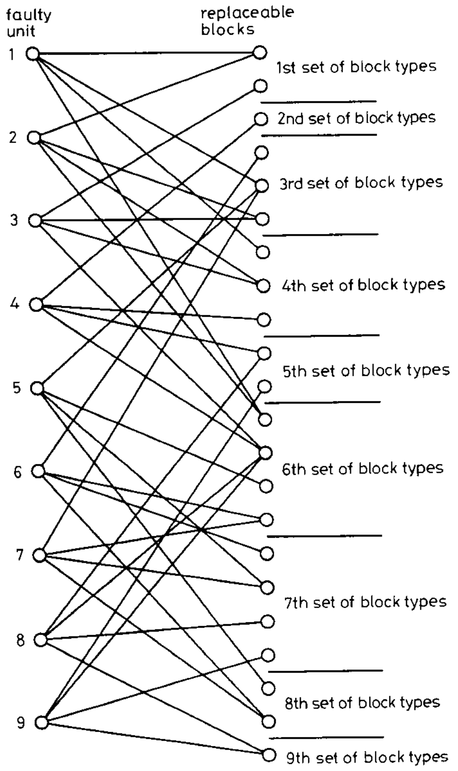

Fig. 8 *Bipartite graph model of the faulty BRMO structure in Fig. 7*

3 Complexity and algorithms for spare allocation

In the following we will discuss the complexity of spare allocation and associated algorithms for SRMO and BRMO only, as spare allocation for SRSO and BRSO is trivial.

3.1 SRMO

Spare allocation in this class is a bipartite maximum matching problem. The complexity of the algorithm for the bipartite matching is $O(|V|^{1/2}|E|)$ [13]. If each edge is assigned a cost, the problem becomes weighted bipartite matching, where a matching with minimum cost is to be found. A weighted bipartite graph can always be made complete by adding nodes and edges to make the two sets of nodes equal in size and all the added edges having a cost larger than any cost in the original graph.

For the weighted matching problem, the complexity of the algorithm is $O(|V|^3)$ for a complete bipartite graph with $2|V|$ nodes by using the Hungarian Method [13]. Before applying the matching algorithm, a check can first be made as to whether the number of spare units is greater than or equal to the number of faulty units; if not, then the system is not reconfigurable. Otherwise, the bipartite weighted matching algorithm is applied to obtain the matching M with minimum cost. If all the faulty units are covered by M, then M is the solution which has a one-to-one relationship between all the faulty units and spare units; otherwise the system is not reconfigurable.

3.2 BRMO

First, consider the restricted case with only two types of block. The authors have shown that the problem of spare allocation for reconfigurable memory arrays with spare rows and columns is a constrained bipartite vertex covering problem and the complexity is NP-complete [3]. This is stated formally in the following lemma:

Lemma: Given a bipartite graph $BG = (A, B, E)$ and positive integers $x_A \leqslant |A|$, $x_B \leqslant |B|$. The problem of determining if there is a vertex cover $V_A \cup V_B$, $V_A \subseteq A$ and $V_B \subseteq B$ such that $|V_A| \leqslant x_A$ and $|V_B| \leqslant x_B$ is NP-complete.

It is easily seen that for the *BRMO* model with exactly two sets of blocks, the spare allocation problem is equivalent to that in the above lemma. For the general BRMO model with more than two sets of blocks, spare allocation is a restricted bipartite dominating set problem with constraints. Vertex covering for the case of two types of block is a special case of this problem. The complexity of this new problem is again NP-complete, which is formally stated in the following theorem [14].

Theorem: Given a bipartite graph $BG = (A, B, E)$, with B partitioned into n sets, $B_1, B_2, \ldots, B_n$, and n positive integers $x_1 \leqslant |B_1|, x_2 \leqslant |B_2|, \ldots, x_n \leqslant |B_n|$, the problem of determining if there is a set $D_1 \cup D_2 \cup \cdots \cup D_n$, which covers all the nodes in A, and $D_1 \subseteq B_1$, $D_2 \subseteq B_2$, $\ldots$, $D_n \subseteq B_n$ such that $|D_1| \leqslant x_1$, $|D_2| \leqslant x_2$, $\ldots$, and $|D_n| \leqslant x_n$, is NP-complete.

The weighted version of the problem in the above theorem is still NP-complete, as the unweighted problem is a special case of the weighted version. If the number of spares is small, an exhaustive search to find an allocation may be applicable. For the case of a reconfigurable design with only two types of block, a branch-and-bound with early-abort algorithm, as well as an approximation algorithm, has been implemented by the authors [3]. In a design with n types of block, the early-abort technique does not apply. However, with some modification, the branch-and-bound technique can still be utilised for moderate size problems. For large problem sizes, the authors have developed a heuristic dominating set algorithm. The algorithm works by allocating spares to cover the nodes in A with degree one. These faults must be replaced first as there is no other choice for them. For those nodes in A that are connected to nodes in B with degree one, spares are allocated to replace nodes in B that cover those nodes in A and have minimum cost divided by degree. Finally, a node in B is selected with minimum cost/degree, and a spare is allocated for it if one is available. The process is repeated until either all the faulty units are replaced, or no spares are left. In the latter case, the structure is not reconfigurable, Fig. 9 shows the details of the heuristic dominating set algorithm, where $S_1, S_2, \ldots, S_n$ represent the number of spare blocks in each block, respectively.

An example spare allocation process for a BRMO scheme with two types of block follows. Fig. 10*a* shows the configuration of a 7×9 rectangular array with two spare rows and three spare columns. R_i indicates row i, and C_j indicates column j. Also shown in the figure are nine faulty cells. There are two pairs of special faulty cells indicated by a line connecting them. If one of the two

Table 1 : Experimental results for BRMO and SRMO

Array size	Number of spare rows	Number of spare columns	Number of faults	BRMO		SRMO	
				repaired	time (s)	repaired	time (s)
64 × 64	2	2	4	yes	0.74	yes	0.97
64 × 64	4	4	32	no	0.98	yes	1.61
64 × 64	8	8	128	no	0.86	yes	1.91
64 × 64	8	8	256	no	0.67	yes	1.61
64 × 64	8	8	512	no	0.40	no	1.40
128 × 128	8	8	16	yes	6.05	yes	7.98
128 × 128	8	8	64	no	4.71	yes	7.72
128 × 128	8	8	256	no	4.24	yes	8.72
128 × 128	8	8	512	no	2.97	no	7.56
128 × 128	20	20	1024	no	1.92	yes	7.40
128 × 128	20	20	2048	no	1.09	no	4.62
128 × 128	24	24	64	yes	7.90	yes	12.78
256 × 256	2	2	16	no	41.32	yes	49.13
256 × 256	4	4	64	no	36.04	yes	51.29
256 × 256	4	4	256	no	29.20	no	50.78
256 × 256	10	10	16	yes	50.53	yes	58.91
256 × 256	16	16	20	yes	52.26	yes	62.94
512 × 512	4	4	256	no	219.15	yes	334.72
512 × 512	18	18	16	yes	423.77	yes	455.43
512 × 512	20	22	4096	no	80.14	no	408.61
512 × 512	32	32	64	yes	400.43	yes	493.11
512 × 512	32	32	8192	no	56.19	no	300.74
512 × 512	128	128	16384	no	98.09	yes	382.69

cells with a connecting line is replaced by a spare, then the other will not need a spare and will function correctly. In memories, for example, this could be a coupling fault in which a *write* operation that affects a state transition in cell x also changes the state of another cell y. By using the BRMO model, this scenario is represented in Fig. 10*b*. The dotted edges represent the effects of coupling faults. By using the algorithm in Fig. 9, we obtain a solution set which includes R_1, R_4, C_4 and C_9 represented by black dots.

```
Algorithm spare_allocation_BRMO:
begin
  for each node $u$ in $A$ with degree one do
  begin
    let $v$ be the node in $B$ which is connected to $u$;
    if $v$ is in $B_i$ and $S_i > 0$ then
    begin
      allocate a spare for $v$ in $B_i$;
      $S_i := S_i - 1$;
      add $v$ in $D_H$;
      delete $v$ and all the nodes in $A$ covered by $v$ and associated
      edges;
      end
  end
  for each node $y$ in $B$ with degree one do
  begin
    let $x$ be the node in $A$ connected to $y$;
    select a node $w$ connected to $x$ in $B_i$ with $S_i > 0$ and minimum
    $c(w)/d(w)$;
    allocate a spare for $w$;
    $S_i := S_i - 1$;
    add $w$ in $D_H$;
    delete $w$ and all the nodes in $A$ covered by $w$ and associated
    edges;
  end
  delete all isolated nodes;
  while $BG$ is not empty and $(S_1 + S_2 + \cdots + S_n) > 0$ do
  begin
    let $v$ be the node in $B_i$ with minimum $c(v)/d(v)$ among all nodes in
    $B$ and $S_i > 0$;
    allocate a spare for $v$;
    $S_i := S_i - 1$;
    add $v$ in $D_H$;
    delete $v$ and all the nodes in $A$ covered by $v$ and associated edges;
  end
  if $BG$ is not empty then
    return fail
  else
    $D_H$ is the repair solution;
end
```

Fig. 9 *Spare allocation algorithm for BRMO structures*

3.3 Experimental results

The algorithms have been implemented in C. Table 1 presents execution times on a Sun 3/60 for both the SRMO and the BRMO schemes for several examples. The examples are for square arrays with spare rows and columns. The program can handle variable sized arrays and general BRMO and SRMO schemes. Although the complexity of the general BRMO problem is NP-complete, the implemented heuristic BRMO algorithm has polynomial time complexity, and therefore it does not always

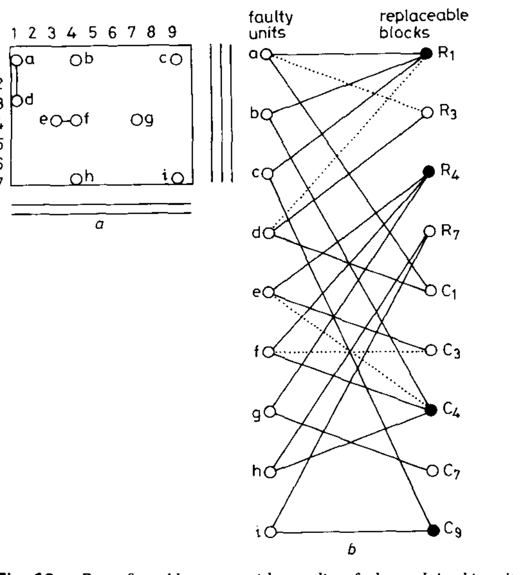

Fig. 10 *Reconfigurable array with coupling faults and its bipartite representation*

find an existing optimal solution. It also runs faster than the matching algorithm for the SRMO scheme for the examples of Table 1. From the table, we can see that for the same problem instance, if the array is repairable under the BRMO scheme, then it is also repairable under the SRMO scheme. However, the reverse is not true. This is due to the higher flexibility of the SRMO scheme in terms of the routing capability.

4 Summary

The main contribution of this paper is to develop a bipartite graph theoretic approach to modelling spare allocation in repairable VLSI structures with dedicated spares. The models describe the relationship between faults and spares. Based on the models, spare allocation is either a matching or a dominating set problem. The complexities of the problems were presented, and the spare allocation algorithms were described. The algorithms have been implemented in C on Sun 3/60 workstations, and example experimental results were also presented.

5 References

1 MOORE, W.R.: 'A review of fault-tolerant techniques for the enhancement of integrated circuit yield', *Proc. IEEE*, May 1986, **74**, pp. 684–698

2 DAY, J.R.: 'A fault-driven, comprehensive redundancy algorithm', *IEEE Des. & Test Comput.*, June 1985, **2**, pp. 35–44

3 KUO, S.-Y., and FUCHS, W.K.: 'Efficient spare allocation for reconfigurable arrays', *IEEE Des. & Test Comput.*, Feb. 1987, **4**, pp. 24–31

4 WEY, C.-L., and LOMBARDI, F.: 'On the repair of redundant RAMs', *IEEE Trans.*, March 1987, **CAD-6**, pp. 222–231

5 LOMBARDI, F., SCIUTO, D., and STEFANELLI, R.: 'A technique for reconfiguring two dimensional VLSI arrays', *Proc. Real-Time Systems Symp.*, December 1987, pp. 44–53

6 HADDAD, R.W., and DAHBURA, A.T.: 'Increased throughput for the testing and repair of RAMs with redundancy'. Proc. Int. Conf. on Computer-Aided Design, Nov. 1987, pp. 230–233

7 HASAN, N., CONG, J., and LIU, C.L.: 'A new formulation of yield enhancement problems for reconfigurable chips'. Proc. IEEE Int. Conf. on Computer-Aided Design, Nov. 1988, pp. 520–523

8 ABRAHAM, J.A., BANERJEE, P., CHEN, C.Y., FUCHS, W.K., KUO, S.-Y., and REDDY, A.L.N.: 'Fault tolerance techniques for systolic arrays', *Computer*, July 1987, **20**, pp. 65–75

9 CHEAN, M., and FORTES, J.A.B.: 'A taxonomy of reconfiguration techniques for fault-tolerant processor arrays', *Computer*, Jan. 1990, **23**, pp. 55–69

10 BANERJEE, P., KUO, S.-Y., and FUCHS, W.K.: 'Reconfigurable cube-connected cycles architectures'. Proc. 16th Int. Symp. on Fault-tolerant Computing, July 1986, pp. 286–291

11 SINGH, A.D.: 'Interstitial redundancy: An area efficient fault tolerance scheme for large area VLSI processor arrays', *IEEE Trans.*, Nov. 1988, **37**, pp. 1398–1410

12 FUCHS, W.K., CHANG, M.F., KUO, S.-Y., MAZUMDER, P., and STUNKEL, C.: 'The impact of parallel architecture granularity on yield', *in* MOORE, W.R., MALY, W., and STOJWAS, A.J. (Eds.): 'Yield modeling and defect tolerance in VLSI' (Adam Hilger, London, 1988), pp. 163–174

13 PAPADIMITRIOU, C.H., and STEIGLITZ, K.: 'Combinatorial optimization, algorithms and complexity' (Prentice-Hall, Inc., New Jersey, 1982)

14 GAREY, M.R., and JOHNSON, D.S.: 'Computers and intractability' (W.H. Freeman and Co., New York, 1979)