Fault Location in Multistage Interconnection Networks with Global and Distributed Control

Ernst G. Bernard, Matthias Sauer and Josef A. Nossek

Institute for Network Theory and Circuit Design, TU Munich
Arcisstr. 21, 80290 Munich, Germany

Abstract

In this paper a procedure is presented for locating single faults in $N \times N$ cube-type interconnection networks with global and with distributed control. Networks with distributed control are assumed to be based on comparator cells with bit-serial architecture. It is shown that fault location requires eight tests for interconnection networks with global control under a functional fault model and four sets of test vectors for interconnection networks with distributed control under the stuck-at fault model, given a specific implementation of the comparator cell. The outstanding feature of this procedure is that the input vectors are independent not only of the network size, but also of the actual fault location. Thus, the fault location procedure can easily be implemented and has low time consumption.

1 Introduction

The complex circuits fabricated with advanced VLSI technology suffer from the susceptibility to physical hardware defects. Yield enhancement through fault tolerance can be provided by reconfiguration techniques for which it is crucial to locate faulty components. Multistage interconnection networks (MINs) are well suited for applying fault tolerance techniques due to their regular and modular topology [1]. A variety of MINs has been proposed and studied in the past years [2], which belong to a class of topologically equivalent networks [3].

In the past years fault location techniques concerning globally controlled INs [4] and locally controlled INs [5, 6] have been studied which use essentially the same $O(\log \log N)$ algorithm. Recently, the algorithm of [4] was improved to have a constant number of tests [7]. A major drawback of these algorithms is the need for analyzing the test results immediately in order to derive further tests. Although our procedure is based on a similar concept, it surpasses previous approaches by the fact that not only the number of tests is independent of the network size, but also the parameters of the tests are independent of the actual fault location. These properties imply simple implementation of the fault location procedure and low time consumption. Similar to other procedures this one is used for off-line fault location. It may be applied after fabrication or every time data transmission can be interrupted.

2 Network architecture

Our procedure is outlined for the generalized cube network (GNC) [8] which serves as a representative for the class of topologically equivalent MINs and it can be easily adapted to any member of that class. Our index notation and the structure of switching elements (SEs) are presented in the following.

The topology of a GCN is depicted in Fig. 1. A GCN for N inputs, where $N = 2^n$, consists of $\log_2 N$ stages with $N/2$ SEs each, which are labelled by s and p, respectively, with $0 \leq s < n$ and $0 \leq p < N/2$. Links are generally indexed by i, with $0 \leq i < N$. N data words form an input vector and are fed to the input links. The input links and the SEs of a stage s are denoted by i^s and p^s, respectively. Furthermore, i^n denotes an output terminal. Since $N = 2^n$, the indices can be expressed by binary numbers with n digits: $i = (i_{n-1}i_{n-2}\cdots i_1 i_0)$.

The data dependencies in a stage s are realized by Perfect Shuffle and Unshuffle permutations [9], which can be represented as cyclic shift left and right operations on the $p-s$ least significant bits of the binary indices of the links, respectively. Thus, the indices of the two inputs of a SE p^s differ exactly by bit $n-s-1$.

For globally controlled INs the SEs are realized by simple 2×2 switches that are controlled by externally generated signals and are assumed to perform either straight or cross connection.

In this paper the SEs of locally controlled INs are assumed to implement a compare and exchange operation (CEO). These elements are denoted in the sequel as CEO-elements and are the constituents of a variety of sorting and merging networks. In particular, replacing the SEs of a

0-8186-5602-6/94 © 1994 IEEE

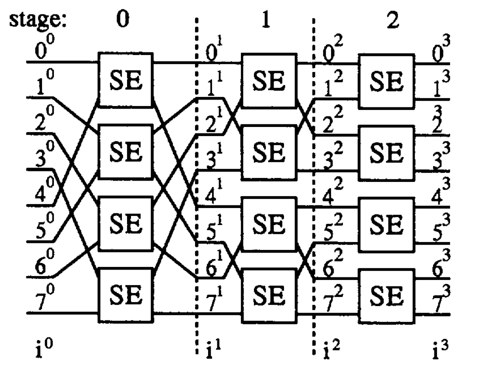

Figure 1: Topology of GCN for $N = 8$.

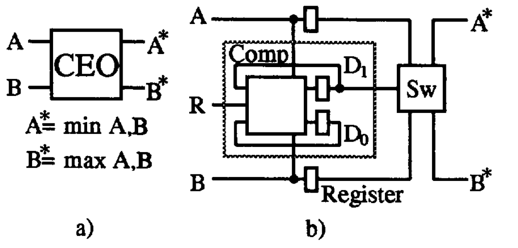

Figure 2: Schematic (a) and structure (b) of CEO-element.

GCN by CEO-elements results in Batcher's bitonic merging network [10]. The inputs and outputs of single SEs as well as CEO-elements are generally referred to as A, B and A*, B*, respectively.

As denoted in Fig. 2a, a CEO-element compares its two input data words A and B and exchanges them, if $A > B$. The CEO-element is usually composed of two functional components (see Fig. 2 b), the comparator (Comp) which investigates if $A > B$, and the switch (Sw) which actually executes the exchange. A well suited data format for this structure is the bit-serial MSB-first format with a word length of b bits per data word.

The comparator is usually implemented as a state machine with two state variables (D_1 and D_0). Its inputs are two bit streams, which consist of the bits A^j and B^j, respectively, $j \geq 0$. These bit streams are subdivided by a reset signal R into data words A_k and B_k, with $k = \lfloor j/b \rfloor$ and $j \bmod b = 0$ for the MSBs.

The comparator examines the bits of two input words A_k and B_k one after another. It remains in its initial state *equal*, as long as $A^j = B^j$. When the first pair of bits with different values occurs, the state machine either assumes the state *greater*, if $A^j > B^j$, or the state *smaller*, otherwise. This state is retained irrespectively of the subsequent values of A^j and B^j, until the reset signal indicates the beginning of a new data word. One of the state variables of the comparator is used to indicate the state *greater* and it controls the switch.

3 Fault location for globally controlled interconnection networks

Following [4], a 2×2 SE of a globally controlled IN can be modelled as a crossbar switch with the 16 functional states depicted in Fig. 3. However, since only the states S_5 (cross connection) and S_{10} (straight connection) are valid ones, the other states are considered as functional faults. The mapping of control signal values to functional states is assumed to be time-invariant. According to [4] every possible functional fault of a single SE and every link stuck-at fault can be detected, if the data words 10 ($= t$) and 01 ($= \bar{t}$), or vice versa, are applied to the two inputs of a SE in both states S_5 and S_{10}. Every single fault within the entire IN can be detected, if each SE in the network gets the inputs t and $\bar{t}$ in both of its states S_{10} and S_5, which is accomplished by the two tests T10 and T5.

For deriving the test vector a for T10, note that every cell gets inputs the binary indices of which differ exactly by the value of one bit. Therefore, one of the indices of the inputs of a SE has an even number of 1's (even parity), the other an odd number of 1's (odd parity). Thus, a can be set up by assigning t and $\bar{t}$ to i^0, if i^0 has even or odd parity, respectively. For the test T5 the test vector a is also sufficient, since all SEs of a stage being in state S_5 results in the data words t and $\bar{t}$ being simply exchanged but the SEs of subsequent stages still get different inputs.

Since under the single fault assumption no fault masking can occur, every fault can be detected at the network output. Depending on whether a faulty output is detected or not the IN is either fault free or the fault can be assigned to one of the following fault classes:

1. One faulty output in exactly one of the tests (S).
2. One faulty output in both tests (2S).
3. Two faulty outputs in at least one test(M).

In the following, a method for locating the faulty SE for each of these fault classes will be derived.

Fault Class S: It is possible to compute a set of links connected to possibly faulty SEs, but it is not possible to determine in which stage the faulty SE is actually located. For specifying the faulty stage two more tests L1 and L2 have to be applied, which are specified as follows:

L1: SEs in stage s are set to $\left\{ \begin{matrix} S_{10} \\ S_5 \end{matrix} \right\}$, if s is $\left\{ \begin{matrix} \text{even} \\ \text{odd} \end{matrix} \right\}$

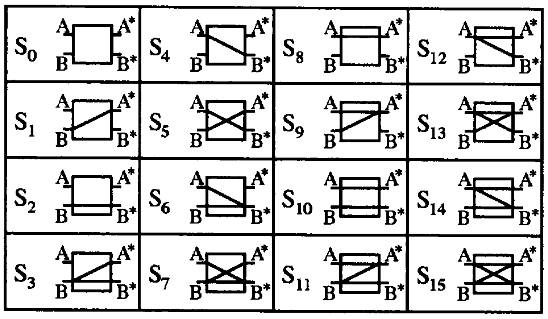

Figure 3: Functional states S_0-S_{15} of a SE.

L2: SEs in stage s are set to $\left\{ \begin{array}{c} S_5 \\ S_{10} \end{array} \right\}$, if s is $\left\{ \begin{array}{c} \text{even} \\ \text{odd} \end{array} \right\}$

The vector a is also a valid test vector for L1 and L2.

Definition 1 A fault path P_X is the set of links $\{i^0, i^1, \ldots i^n\}$ that forms a path from an input i^0 to an output i^n while the setting of the IN is according to $X \in \{\text{T10}, \text{T5}, \text{L1}, \text{L2}\}$.

Proposition 1 *Given two intersecting paths P_T and P_L with $T \in \{\text{T10}, \text{T5}\}$, $L \in \{\text{L1}, \text{L2}\}$ and $i^s \in P_T \cap P_L$, then the two implications $\forall i^{s+1}[i^{s+1} \notin P_T \cap P_L] \Rightarrow \forall i^{s'}[i^{s'} \notin P_T \cap P_L \wedge s' > s]$ and $\forall i^{s-1}[i^{s-1} \notin P_T \cap P_L] \Rightarrow \forall i^{s'}[i^{s'} \notin P_T \cap P_L \wedge s' < s]$ hold.*

For proof consider that the indices of the output links of p^s differ exactly by bit $n - s - 1$, which does not hold for the inputs of SEs in all successive and preceding stages.

Lemma 1 *Given two paths P_T and P_L, $T \in \{\text{T10}, \text{T5}\}$ and $L \in \{\text{L1}, \text{L2}\}$, then $0 \leq |P_T \cap P_L| \leq 2$ holds.*

Proof: Assume p^s is set to the same state for both settings corresponding to T and L. Then, $\exists i^s, i^{s+1}\, [i^s, i^{s+1} \in P_T \cap P_L]$. Since p^{s-1} and p^{s+1} are set to different states for the two settings, i^s and i^{s+1} are the only elements of $P_T \cap P_L$ (Proposition 1). If for some $i^s \in P_T \cap P_L$ there is no SE p^s that is connected to this link i^s and has the same state for both settings, $|P_T \cap P_L| = 1$ holds.

Clearly, since the number of paths P_T which have at least one common link with a path P_L is $\lfloor n/2 \rfloor + 1$, there are $N - \lfloor n/2 \rfloor - 1$ paths P_L, which a path P_T has no common link with. □

Theorem 1 *A fault of fault class S can be located by applying the four tests T10, T5, L1, L2.*

Proof: Let i^n be the faulty output of T10 or T5 and i'^n be the faulty output of L1 or L2. Also, let $i^n \in P_T$ and $i'^n \in P_L$. $P_T \cap P_L$ yields a set of two links (see Lemma 2) which are connected to exactly one SE that produces a fault for either S_{10} or S_5. □

Fault Class 2S: If a fault was detected in both test phases T10 and T5, one link can be determined which is common to both fault paths P_{T10} and P_{T5}. The problem is to determine in which of the SEs adjacent to this link the fault is actually located.

Proposition 2 *Two paths P_{T10} and P_{T5} have at most one link i^s in common.*

The proof is analogous to Proposition 1.

The following theorem makes use of the fact that if a faulty SE has equal inputs, its output will be correct unless an output terminal has an open fault, i.e. no input is connected to this output terminal and it behaves like a stuck-at fault. We use four additional tests S1-S4, which are specified by all SEs being either in S_{10} (tests S1 and S3) or S_5 (tests S2 and S4) and test vectors that provide SEs in stage s with equal inputs, if s is even (S1 and S2) or odd (S3 and S4).

Theorem 2 *Faults of class 2S can be located by the tests T10, T5 and S1-S4, unless the fault is a link stuck-at fault or an open link for both states of the SE.*

Proof: Due to Proposition 2 only a link i^s can be determined, though actually some SE either in stage $s - 1$ or s may be defective. However, if the fault detected for state S_{10} is not an open fault, one of the tests S1 and S3 yields a correct response and the fault can be located in the stage the SEs of which have equal inputs. The same applies for the fault for state S_5.

Thus, all faults of class 2S can be located independent of the actual location except for open faults for both states S_{10} and S_5. Then the faulty behavior is the same as for a link stuck-at fault on the common link i^s. □

Fault Class M: At least in one of the tests T10 and T5 faults are detected on two output terminals i^n and i'^n.

Theorem 3 *Faults of class M can be located by T10 and T5.*

Proof: The indices of the two incorrect data words at the output of stage s, which contains the faulty SE p^s, as well as stage n differ exactly in bit i_{p-s}. Thus, $s = n - \log_2 |i^n - i'^n| - 1$. The index of the defective SE p^s can be determined with given output i^n (or i'^n) and the stage index s as follows: $p^s = \lfloor i^n/2^{n-s} \rfloor \cdot 2^{n-s-1} + i^n \bmod 2^{n-s-1}$, if the double fault was detected by T10, or $p^s = (\lfloor i^n/2^{n-s} \rfloor - 1) \cdot 2^{n-s-1} - i^n \bmod 2^{n-s-1} - 1$, otherwise. □

By applying all eight tests derived above (i.e. T10, T5, L1, L2, S1–S4), every faulty SE in the GCN can be

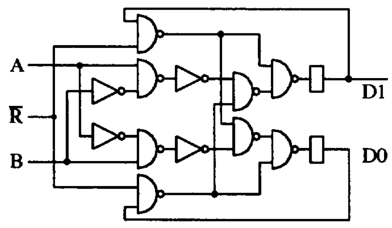

Figure 4: Implementation of comparator.

detected independent of the network size and the actual fault location, unless the fault type is the one mentioned for class 2S.

In the following section these results are used to develop a fault location procedure for locally controlled INs.

4 Fault location for locally controlled interconnection networks

Due to the complex structure and a variety of possible implementations, it will not prove fruitful to devise a general model for CEO-elements as for globally controlled SEs. The effects of CEO-cell faults cover a range from routing faults (wrong exchange decision) to corruption of data words. Furthermore, the CEO-elements have to be controlled solely by the input vectors.

As an illustrative example for the fault location procedure for locally controlled INs a common implementation of the comparator is used [11], see Fig. 4. First, it is assumed that faults occur only in the comparator, but later on it will be shown that all switch faults are also detected by the comparator tests.

For the derivation of the test patterns the stuck-at fault model at gate-level was used. Simulation results showed that this fault model is adequate, even in the presence of typical CMOS defects. Thus, the following conditions for the inputs of a CEO-element have been derived for which all comparator faults are detected:

$$\begin{aligned}
C1 &: A_k < B_k \wedge \forall j[A^j = B^j = 1 \wedge j \bmod b = 0] \\
C2 &: A_k > B_k \wedge \forall j[A^j = B^j = 1 \wedge j \bmod b = 0] \\
C3 &: A_k < B_k \wedge \exists j[A^j > B^j \wedge j \bmod b > 0] \\
C4 &: A_k > B_k \wedge \exists j[A^j < B^j \wedge j \bmod b > 0] \\
C5 &: A_k < B_k, A_{k+1} > B_{k+1} \vee \\
&\qquad A_k > B_k, A_{k+1} < B_{k+1}
\end{aligned}$$

Clearly, $\lfloor j/b \rfloor = k$ must hold for C1–C4. Condition C5 is met implicitly when the conditions C1–C4 are fulfilled. Therefore, for testing all comparators in a locally controlled IN the conditions C1–C4 have to be met at the inputs of every CEO-element. For C1 and C2 this is satisfied by sorted and reversely sorted input vectors, respectively, where the MSBs of data words are set to 1.

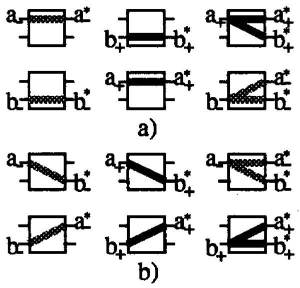

Figure 5: Passing of fault by fault free CEO-cell.

All test vectors for locally controlled INs can be generated by the node labelling procedure *n_cube_assign* presented in [12]. It works on a binary n-cube, a graph on which the GCN can be mapped by associating nodes of the graph with data links and its edges with the CEOs of the IN. The procedure visits all nodes in the order of a breadth-first-search, starting from a root node with an initial label and giving each node a label which depends on the label of its predecessor. These labels are used as the data words of the test vectors. The labelling function has to be chosen according to the problem at hand. For instance, for conditions C3 and C4 the labelling function is $lab_{new} = 2 \cdot lab_{old}$, the initial label is 1 and the root nodes are 0 and $N-1$, respectively.

The tests for conditions C1-C4, denoted by CT1-CT4, can detect every single comparator fault in a locally controlled IN. A faulty CEO-element will output one or two incorrect data words (denoted by a^* and b^*) which may be smaller or greater than the correct ones, denoted e.g. by $a^*_+ > A^*$. Due to Proposition 1 incorrect data words are only compared with correct ones, i.e. each CEO-element can have at most one incorrect input. An incorrect data word is either passed by a fault free CEO-element as in the fault free case, or conditionally a second incorrect data word is produced as depicted in Fig. 5. Hence, a fault *tree* with possibly more than one incorrect data word per stage is generated rather than a fault *path*. Since incorrect inputs which are smaller (greater) than the correct ones result in incorrect outputs which are also smaller (greater) than the correct ones, a fault tree can be reduced to a fault path using the knowledge about fault free behavior of the CEO-element and the rules depicted in Fig. 5. Thus, fault location for locally controlled INs can be achieved similarly to globally controlled INs.

Theorem 4 *Every faulty comparator in the CEO-elements of a locally controlled IN can be located by applying four sets of input vectors.*

Proof: 1.) If a faulty CEO-element has two incorrect outputs one of which is greater, the other smaller than the corresponding data word in the fault free case, two fault paths can be derived and the fault located equivalently to class M for globally controlled INs.

2.) A comparator fault is equivalent to fault class S if only one fault path can be obtained from CT1-CT4. A second set of tests CL1-CL4 (similar to L1 and L2 for globally controlled INs) is required for fault location. For these tests complementary conditions are met in successive stages, e.g. for CT3 conditions C3 and C4 are met in stages with even and odd indices, respectively. The test vectors can be generated by *n_cube_assign* by simply using a new root node.

3.) A comparator fault is equivalent to the fault class 2S if two disjoint fault paths are obtained from CT1-CT4. Since in the worst case all four tests CT1-CT4 detected faults, the location problem can only be tackled by applying equal inputs to the CEO-elements, since equal (and correct) outputs are obtained at the output terminals irrespective of the faulty behavior of the comparator. According to the tests S1-S4, equal inputs have to be applied to the CEO-elements of even and odd stages alternately by two sets of tests CSO1-CSO4 and CSE1-CSE4 while the CEO-elements in other stages get the normal test inputs. For the generation of test vectors with *n_cube_assign* the labelling function has to be modified in order to accomplish equal data words in the respective stages.

This shows that applying four sets of four tests each is sufficient to locate every comparator fault. □

Corollary 1 *Switch faults and transmission link faults are located with the set of input vectors for locating comparator faults.*

Proof: For detecting transmission link stuck-at faults and switch faults according to the functional fault model of Sec. 3 the bits of a pair of test words have to meet the conditions $0 = A^j < B^j = 1$ and $1 = A^{j'} > B^{j'} = 0$, with $j \neq j'$ and $\lfloor j/b \rfloor = \lfloor j'/b \rfloor$. Thus, the tests CT3 and CT4 are equivalent to T10 and T5. Similarly, the tests CL3 and CL4 correspond to L1 and L2, and the tests S1-S4 are covered by CLO3, CLO4, CLE3, and CLE4. □

Thus, for locating single defective components in a locally controlled IN 16 tests are sufficient independent of the network size and the actual fault location.

5 Conclusion

In this paper we first presented a procedure for locating single faults for any member of a class of globally controlled MINs. We showed that eight tests are sufficient for locating single faults, independent of the network size and the actual fault location.

These results can be adapted to locally controlled interconnection networks, the switching elements of which are implemented as comparator cells. Four sets of test vectors have to be applied to the network in order to achieve fault location, again independent of the network size and the actual fault location.

This unified approach to fault location in MINs has the advantage of being independent of the routing scheme except for the data words necessary for detecting faults. Since the tests are independent of the network size and the actual fault location, the implementation of our procedure is less intricate and a significantly lower time consumption can be achieved compared to other approaches so that even on-chip implementations are possible.

References

[1] D. P. Agrawal. Testing and fault-tolerance of multistage interconnection networks. *IEEE Computer*, 15(4):41–53, 1982.

[2] Tse-Yun Feng. A survey of interconnection networks. *IEEE Computer, special issue*, 14(12):12–27, 1981.

[3] Chuan-Lin Wu and Tse-Yun Feng. On a class of multistage interconnection networks. *IEEE Trans. Comp.*, C-29(18):694–702, 1980.

[4] Tse-Yun Feng and Chuan-Li Wu. Fault-diagnosis for a class of multistage interconnection networks. *IEEE Trans. Comp.*, C-30(10):743–758, 1981.

[5] N. J. Davis, W. T.-Y. Hsu, and H. J. Siegel. Fault location techniques for distributed control interconnection networks. *IEEE Trans. Comp.*, C-34(10):902–910, 1985.

[6] Willie Y-P. Lim. A test strategy for packet switching networks. In *Proc. Int. Conf. on Parallel Processing*, pages 96–98, 1982.

[7] W.-K. Huang and F. Lombardi. On the constant diagnosability of baseline interconnection networks. *IEEE Trans. Comp.*, 39(12):1485–1488, 1990.

[8] Howard J. Siegel and S. Diane Smith. Study of multistage SIMD interconnection networks. In *Proc. 5th Ann. Symp. on Comp. Arch.*, pages 223–229, 1978.

[9] H.S. Stone. Parallel processing with the perfect shuffle. *IEEE Trans. Comp.*, C-20(2):153–161, 1971.

[10] K.E. Batcher. Sorting networks and their applications. *Proc. AFIPS 1968 SJCC*, 32:307–314, 1968.

[11] U. Kleine. Novel sorter architecture for image processing rank order filters. *Electronics Letters*, 23(1):45–46, 1987.

[12] E. G. Bernard, M. Sauer, and J. A. Nossek. Fault detection in bitonic merging networks. In *Proc. ECCTD'93*, 1993.