JOURNAL OF

ELECTRONIC TESTING

Theory and Applications (JETTA)

Editor in Chief: Vishwani Agrawal

Volume 2, Number 1, March 1991

KLUWER ACADEMIC PUBLISHERS

CODEN JTTAER
ISSN: 0923-8174

JOURNAL OF ELECTRONIC TESTING:

Theory and Applications

(JETTA)

AIMS AND SCOPE

The *JOURNAL OF ELECTRONIC TESTING: Theory and Applications* is an international forum for the dissemination of research and application information in the area of electronic testing. The papers for publication in the *JOURNAL OF ELECTRONIC TESTING* are selected through a peer review to ensure originality, timeliness and relevance. The journal provides archival material, and through its quick publication cycle, brings recent results to researchers and practitioners.

The *JOURNAL OF ELECTRONIC TESTING: Theory and Applications* is the only research journal devoted exclusively to electronic testing.

JOURNAL OF ELECTRONIC TESTING: Theory and Applications, 2, 77-88 (1991)

Boundary Scan Test, Test Methodology, and Fault Modeling

FRANS DE JONG
Philips, CFT-Automation, Box 218, 5600 MD Eindhoven, The Netherlands

JOSÉ S. MATOS AND JOSÉ M. FERREIRA
INESC, Portugal

Received February 5, 1990. Revised September 11, 1990.
Editor: R.G. Bennetts

Abstract. The test technique called "boundary scan test" (BST) offers new opportunities in testing but confronts users with new problems too. The implementation of BST in a chip has become an IEEE standard and users on board level are the next group to begin thinking about using the new possibilities. This article addresses some of the questions about changes in board-level testing and fault diagnosis. The fault model itself is also affected by using BST. Trivial items are extended with more sophisticated details in order to complete the fault model. Finally, BST appears to be a test technique that offers a high degree of detectability on board level, but for diagnosis, some additional effort has to be made.

Key words: boundary scan test, BST-net, diagnosis, fault modeling, PCB testing, test-pattern generation.

1. Introduction

When using the boundary scan test (BST) in a modern digital design, the accessibility for testing the printed-circuit board (pcb) improves. In this article, boards are assumed to have full BST, which means that all chips have the BST capability. The test patterns to test the board are generated according to a fault model.

Checking of the interconnects between devices on a full BST board is achieved through an imposed flow of signals, through the Test Access Port (TAP), as digital values are exchanged between the BST cells associated with all interconnected pins. The BST infrastructure will act as the electronic bed-of-nails replacement that will allow access (controlling and observing) to the values at the nodes of the board under test. This means that the PCB is accessed by virtual pins that, however, are only effective if the PCB under test is powered. While normal bed-of-nails fixtures give the possibility to check for shorts without powering the PCB, this is impossible when using BST while the BST logic on the PCB has to work to enable testing. Some faults can therefore be fatal before they are found. Once successfully started, testing with BST requires some rethinking on the fault model and the test patterns related to this model in order to be able to make a detailed diagnosis of the faults encountered. The remaining test nodes, that is, functional I/O, connectors, and test spots are controlled in synchronization with the BST signals by means of normal tester techniques, using real pins, or the remainder will be tested functionally.

The BST boards and BST facilities mentioned in this article conform to the JTAG V2.0 BST architecture [1] that was the basis for the IEEE standard 1149.1 [18].

2. The BST-Net

Testing PCBs using BST refers primarily to structural testing. This means generally that testing is checking for correct connections between right components. Using a bed-of-nails technique, interconnects are tested from nail to nail and the components are checked to be alive with a minimal set of "functional" vectors. Using BST, the virtual pins have moved from the copper track or solder joint area to the inside of the IC.

A physical outline of a net using BST shows that not only the board interconnect is involved, but also inner parts of the IC, such as bonding wires and the I/O buffers (the path between A and B in figure 1). Concluding that an interconnect is faulty will thus not necessarily imply that the board interconnect is faulty. The exact location of a detected fault on this BST-net is between an output BST-cell and an input BST-cell. It is

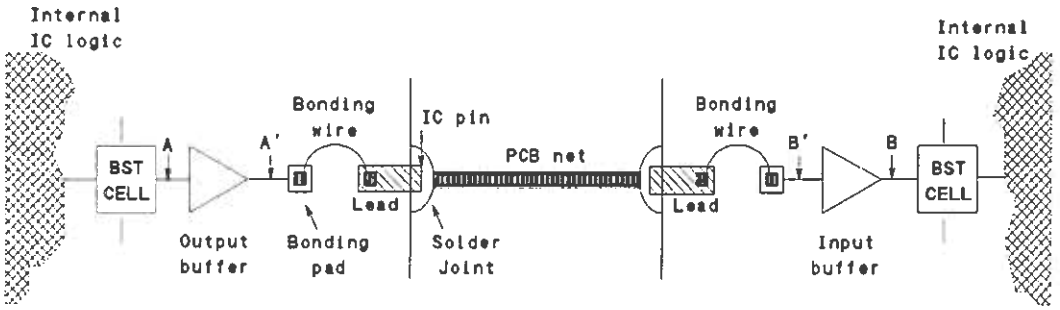


Fig. 1. The BST-net, a physical connection from BST-cell to BST-cell.

obvious that testing with BST gives only diagnostic possibilities to the level of the BST-net connection that is established between the BST-cells. Nets with more than one receiving input can give better diagnostics (i.e., a faulty input can be found). Multiple (driving) outputs can give some relief if the fault occurs at one of the outputs of an IC, provided that the faulty output can still be switched to the high-impedance state.

Two remarks can be made, concerning figure 1:

- It can be seen that path A–B between the two BST cells includes two different technologies. The I/O buffers are IC technology, while the interconnections (path A'–B') are just a galvanic connnection technology.
- Testing the path between points A and B also implies that potential handling hazards occurring in the chip's boundary area, are tested. The two most important handling defects taken into account are:
  a. broken (loose) bonding wires due to vibration or temperature shock (mechanical stress), and
  b. blown buffers due to static discharge (electrical stress).

Possible faults that can be present on a BST-net, that all might look the same to the tester, are given in table 1. This table presents nondistinguishable faults because the reaction of every fault to a given stimulus can be the same, that is, provided that the BST infrastructure itself was found to be correct.

This definition of the BST-net gives rise to some adaptation of the test methodology. As stated before, BST is specially suited for structural testing (in-circuit testing). However, while the accessible nodes are now placed inside the chips, it is possible to test the interconnections without having to deal with the core logic of the ICs. This means that testing a PCB can be fully focused on checking the integrity of the interconnections, which also simplifies the test-pattern generation.

3. Hierarchical Structures and Multiple Taps

The hierarchical nature of the BST technology implies that some remarks are made, concerning the various levels. Although more levels of hierarchy could be possible, it seems reasonable that the following three levels are considered.

- the IC level
- the board level (a set of ICs)
- the system level (a set of boards)

At the IC level, some proposals have been made to extend the BST technology to within the IC, such as in

Table 1. Possible faults on a BST-net.

1.	Stuck-at fault in the output of BST-cell.
2.	Stuck-at fault in output buffer.
3.	Loose or broken bonding wire at output.
4.	Bonding wire shorted (e.g. with neighbour bonding wire).
5.	Bad (cold or none) solder joint at output.
6.	Open or shorted copper track.
7.	Bad solder joint at input.
8.	Bonding wire short at input.
9.	Bonding wire broken at input.
10.	Stuck-at fault in input buffer.
11.	Stuck-at fault in input of BST-cell.

Beenker et al. [2] and van Riessen et al. [3], so that IC testability may be improved by partitioning of the internal logic, thus isolating modules at a lower level. The BST architecture can easily accommodate this methodology to enhance testability, while the approach and test philosophy are the same.

At board level, more than one scan chain (and accordingly more TAPs) can exist according to the JTAG standard; this is also studied by Wang et al. [4]. The existence of more than one scan chain will imply a more sophisticated test equipment, since various data (TDI, TDO) and control (TMS, TCK) lines may have to be controlled simultaneously. Another solution can be different scan paths controlled by a single board-level BST controller [14]. An immediate advantage of the multiple TAP method is the possibility of parallel processing of various test paths, which will result in a faster test execution. So far, no reference has been made to the possibility of board self-test. This topic is not relevant from the test methodology point of view, seen at board level, since the topic directly concerns the implementation of the framework of algorithms that is to be developed for full BST. Whether board testing is done by on-board residing firmware, or by externally applied sets of vectors, the same methodology can be used. Whatever solution is implemented, the basic methodology for testing a board using BST will not change.

This situation may differ at system level. In fact, if all system boards have been proved to work properly, then the remaining task is essentially to perform a test of all interconnects between the various boards, then follow with a limited functional test—but this time at system level. In theory, connecting all the board-level boundary scan paths together would provide a system-level path capable of testing the board-to-board interconnects. Such a scheme would require a system-level BST controller (not defined in the standard); but, in practice, the system-level scan path would quickly become too large. The fact that testing of the various boards should proceed independently, requires that some sort of isolating capability must be available. This could be implemented through chips that interface each board to the common test lines of a system bus as described by Texas Instruments [10] (but defining test lines in an existing system bus is another problem that has to be solved, as mentioned by Vining [14]). Here the statement holds that multiple scan-chain capabilities do not change the test methodology using full BST at system level, but the standard does not describe the means necessary for a system-level connectivity test.

Looking carefully at all this, the idea of using BST at different levels of hierarchy is no problem, but the integration of the different levels to a single BST test approach seems to be not as straightforward. Also integrated test-preparation tools for IC, board, and system levels are not available.

4. Fault Modeling for Full BST

A fault model is a first assumption upon which any fault-oriented test methodology is based. The basic objective of a fault model is to provide a translation mechanism, through which real faults are converted to a more adequate representation scheme. Adequacy, in this sense, refers essentially to the two following aspects:

- the completeness of the translation mechanism;
- ease of computational use of the new representation scheme.

It should be noted that an improvement in one of these aspects will frequently lead to a degradation in the other as stated by Hayes [9]. A good example to illustrate these comments is the well-known single stuck-at fault model used for ICs. Although the stuck-at model is known to be limited (usually only single stuck-ats are considered), the model is still regarded to be a firm base for test-pattern generation algorithms. Enhancements are still developed as seen in Fujiwara [12], and the stuck-at model is in use as seen in Kirkland and Mercer [13].

4.1. Manufacturing Faults

At board level, several types of manufacturing faults can be identified, as seen in table 2. Some of the listed assembly faults (shorts, opens, etc.) are also common to the IC level, but possibly with different rates of occurrence. Other faults (wrong component, backward component, etc.) do not find any correspondence at IC level. Recall also that the IC test is generally a go/no-go test, while board testing requires not only fault detection, but also fault diagnosis so that the board can be repaired. The single stuck-at fault model does not seem an adequate representation scheme for these kind of PCB faults. Therefore, the list of failure mechanisms for board assemblies and their related fault effect must be studied—see table 2.

Studies in this field by Factron [5], Bennetts [11] and Francese [15], have shown (assuming reduced human

Table 2. Board assembly failure mechanisms and their logical effect.

FAILURE MECHANISM (cause)	FAULT (logical effect)
STATIC FAILURES	
- solder splash between adjacent pins (tracks) - connection between two closely routed nets	- short
- no solder joint - broken net - missing component - defective IC bondings	- open
- wrong board version - wrong component - wrong orientation - defective IC buffers	- short/stuck at/opens (this group might look as dynamic failures)
DYNAMIC FAILURES	
- bad solder joint	- intermittent open
- temperature sensitivity - voltage variations	- intermittent faults

error rates and existing incoming inspection) that the most representative board faults are opens and shorts (between signal nets, or between signal nets and power lines). Various works have been done on deriving a fault model for this situation, such as Wang et al. [4], Wagner [6] and Hassan et al. [7].

4.2. Technology Dependency

Considering the digital nature of the BST methodology—the derived fault model must be able to translate physical faults to logical values. Having this as an assumption, several questions can be put, such as the following:

- How does an open translate to a logical value?
- How does a signal-net to signal-net short (bridging) fault translate to a logical value?
- How do signal nets react, when shorted to power nets?

The answers to these questions imply that a fault model may be technology dependent.

4.3. Opens and Technology

A floating input resulting from an open interconnect may be considered as a logical 1 for TTL technology. However, this is not applicable for CMOS technology where the acquired value is random.

The results for different technologies, when the input is left open are illustrated in table 3.

Table 3. The result of a floating input for various technologies.

TTL	CMOS	HCMOS	ECL	BICMOS
Read as a '1'	Random value	Random value	Read as a '0'	Depending per case; mostly TTL like

4.4. Shorts and Technology

The same problem will appear for the short type of faults. In fact, two (or more) shorted nets, where IC outputs at 0 and IC outputs at 1 coexist, will have the connected IC inputs updated to a value that will depend on several factors. If the IC outputs at 0 present a lower impedance than the outputs at 1 in this shorted situation, it is expected that the 0 will prevail, and thus all inputs will be updated to 0 (so called *and-shorts*). The opposite will lead to all inputs being updated to 1 (so called *or-shorts*). It may also happen that the resulting impedance values are not far from one another, and then it will not be possible to foresee to which value will the inputs be updated. This is obviously technology dependent (TTL IC outputs present a lower impedance when at 0 than when at 1, but this is not true for CMOS ICs). Moreover, even when it is known that one logical state presents a lower impedance than the other, the resulting logical value will definitely be influenced by the number of outputs with this value, and the number of outputs at the opposite value. It thus may theoretically happen that a short between TTL outputs will result in a logical 1, due to the low impedance effect of parallel outputs at 1, even though there could be an output at 0 that cannot sink enough current (or even might burn out). So, short faults also translate to logical faults in a technology-dependent manner, and additional care should be taken because the result may be changed by the number, and the state, of shorted outputs.

Shorts between signal nets and power nets should be considered as a special case. According to the previous comments, and considering the low impedance of the power nets, the resulting value will be 0 in a short to ground, and 1 in a short to the positive supply line. Since the power nets are the only nets in a full-BST board that do not have associated BST cells [1], shorts between signal nets and power nets should be regarded as the general stuck-at type of faults. It can thus be concluded that shorts between signal nets and power nets, as a special case of a short, translate to stuck-at faults in a technology independent manner. The diagnosis on this type of fault will result in the stuck-at type of fault.

Finally, the logical fault model to be derived must therefore consider the following fault types:

- stuck-open (floating inputs updated to 1, or 0, depending on each case);
- shorts (or-shorts, where outputs at 1 prevail, or and-shorts, where outputs at 0 prevail); and
- stuck-at (0 or 1).

Randomly updated inputs (such as certain types of floating inputs, or shorts where the resulting value is neither 1 nor 0, or even intermittent and/or dynamic faults) will translate to one of the above cases. It will happen that this shows the lack of completeness of the fault model and readings will result in a sometimes good, sometimes bad result. Also DFT measures could be used to avoid the occurrence of these random situations (input buffers could be designed such that a predetermined value would be acquired when the input is left floating, etc.).

4.5. Multiple Fault Detection

Concerning the IC-level single stuck-at fault model, it should be recalled that only single faults are considered, because there is no time to consider all possible multiple faults. Also low values of controllability and observability at many internal nodes would make this IC testing too complex. This is really not the case with the BST methodology, since there is total controllability and observability at all nodes (considering that nodes are IC pins, and that we are talking of full-BST boards), and thus there is no reason to consider only single faults. Because of this full and direct accessibility, we should reconsider the need to examine all multiple-fault situations.

The amount of computer time to analyze all situations will still be excessive, of course, but even when the analysis time seems reasonable, there is no advantage in considering situations that do not add real information to the diagnosability. The question that arises is thus whether all possible multiple faults should be considered, or whether a subset of these faults is able to provide identical results or at least match the practical requirements for accuracy of fault location. In fact, it could be argued that there is no need to consider shorts between more than 2 nets, since any of these could be found as the summation of multiple 2-net shorts. The following question should then be studied: What reductions can be made on the total number of faults, for the various types (stuck-at, shorts, stuck-open), such that the accuracy of the diagnosis is not significantly affected? It is now of importance to realize that multiple shorts or opens at board level have to do with physical effects such as solder splashes and local damages. One of the major items that influences diagnosability is the restricted area of a fault cluster. So while repairing a 2-net short, the total multiple net short is localized and can be repaired. Multiple opens due

to a scratch can also be repaired at once. This especially points to the meaning of detecting faults in nets that do not interfere with each other with respect to fault detectability (noninterfering nets) and in a later stage (i.e., study on diagnosis) this will lead to the definition of *fault-locality* and/or *fault-cluster* information. It can be seen that this requires knowledge of the board layout regarding net locations and even knowledge about type and frequency of occurrence of faults.

5. The Fault Model and Diagnosability

As far as our board-level fault model is concerned, there is no reason to consider only single occurrences of stuck-at faults. The fault model in discussion (opens, shorts, and stuck-at) can handle detection of all the faults as long as the technology permits detection. The main question is whether the fault model permits finding test vectors that enable a clear diagnosis for those faults. As far as full BST is concerned, multiple faults can be diagnosed fully if those faults occur in noninterfering nets. If multiple faults appear on interfering nets, the result pattern will be influenced and a clear diagnosis on the exact type of fault might be impossible. This means that no problem exists in diagnosing a short between nets a and b and at the same time finding net c to be stuck at 1, but it would be difficult to diagnose exactly what was wrong if nets a and b were shorted and net a was open at the same time.

In general, the fault model allows detection of all faults, even if the interfering faults are present, but diagnosability will be difficult. A drawback of the translation mechanism (i.e., faults translate to digital values) is that it is not always possible to diagnose which fault or what type of fault is responsible for the faulty digital pattern found.

5.1. About Shorts

Concerning the short type of faults, and for fault detection purposes, it is acceptable to consider shorts between 2 nets only (since a short between more than 2 nets and multiple 2-net shorts on the same set of nets will always be detected as a superset of single 2-net shorts). In fact, and considering a board with n nets, the total number of k-net shorts will be generally given by "combinations k out of n," noted as C_k^n. Expressed in a formula:

$$\text{Total number of 2-net shorts} = \frac{n!}{(n-k)! \cdot k!}$$

Here n denotes the total number of nets and k the number of nets shorted. For 2-net shorts, this formula will be C_2^n which reduces to $n(n-1)/2$. In figure 2, where four nets are shown with all possible 2-net shorts, this gives C_2^4 which results in 6 possibilities.

When the total number of shorts between any number of nets (from 2 to n) is considered, then this number will be a summation of all k-net posibilities, given by

$$\text{total number of shorts} = \sum_{i=2}^{n} \frac{n!}{i! \cdot (n-i)!}$$

Here the results for each value of i correspond to the total number of possible shorts between i nets (actually, the variable i replaced the constant value of k).

The difference on the resulting number of possible faults, for various values of n, is shown in table 4. The results in this table show that for any reasonable number of nets on a board, the total number of shorts between any number of nets is much larger than the total number of 2-net shorts.

Table 4. Comparison between two- and multi-net shorts.

n	$\frac{n(n-1)}{2}$	$\sum_{i=2}^{n} \frac{n!}{i!*(n-i)!}$
5	10	26
10	45	1013
20	190	4339
50	1225	>> 10e6
100	4950	...
nets	2-net shorts	multi net shorts

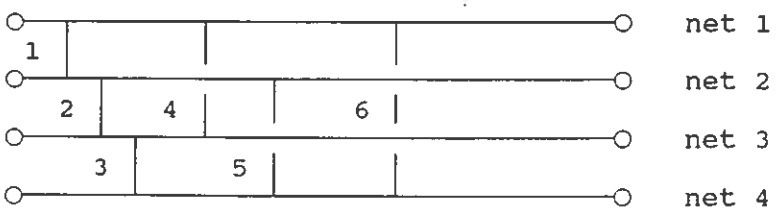


Fig. 2. All 2-net shorts in a group of four nets.

For purposes of complete fault diagnosis, all shorts between any number of nets will have to be considered, but in practice, because of the big numbers and the lower diagnostic value, it is not economical to process all short possibilities.

5.2. *About Opens*

Concerning the stuck-open faults, a similar situation exists. The total number of stuck-open faults in a given board practically depends not only on the number of (BST-) nets, but also on the number of nodes (IC pins) on each net.

Moreover, a net between two nodes may be broken at several different locations at the same time. The detection of multiple opens in a single connection is not possible without additional test points placed between fault locations. It thus seems reasonable that only those situations corresponding to a net being broken at a single location are considered, since diagnosis down to this level is generally considered enough. This means that a stuck-open fault will correspond to a partition of a net in just two subnets.

For a net connected to more than two nodes, the number of faults (i.e., the number of possible partitions in two subnets) that may exist due to an open connection is built up by counting the number of possibilities of opens per type of the resulting net structure. A simple example for a net with p nodes will be given first:

1. Every node can be isolated from the net, which gives p possible faults.
2. Every node can stay in connection with one other node and become isolated from the net as a group. This gives C_2^p faults (a similar formula as used with the 2-net shorts).
3. Every node can stay in connection with two other nodes and be isolated from the remainder of the original net. This results in C_3^p faults.
4. Etc. until all possibilities have been counted.

If studied for $p = 5$, the third step is already too much. A net with five nodes can be split in a group with two and a remainder of three nodes respectively. The other solution, a group of three, with a remaining group of two nodes is exactly the same (step 3). And this also holds for the following numbers. Therefore counting will stop halfway the number of nodes. Put in another way it is easy to count from 1 to $(p - 1)$ and divide the result by 2. As a formula:

$$\text{total number of single opens} = 0.5 \cdot \sum_{i=1}^{p-1} \frac{p!}{i! \cdot (p - i)!}$$

In this formula, the i denotes the number of nodes that is grouped and cut off from the net by the open. The highest value to evaluate is $(p - 1)$ because if all nodes are separated, no remaining net exists, and so no open is found.

This formula can be illustrated by an example, such as the one shown in figure 3. This figure is a schematic diagram of two outputs connected to two inputs. The total number of opens results from the physical layout.

The following definitions are made for figure 3:

output nodes: [0, 1] (represented as a black symbol) "output" refers to the output of the IC that drives the net, that is, source.

input nodes: [2, 3] (represented as a white symbol ○) "input" is the IC's input that reads information from the net, that is, sink.

For this case, by filling in the formula, the total number of opens will be

$$0.5 \cdot \left(\frac{4!}{1! \cdot 3!} + \frac{4!}{2! \cdot 2!} + \frac{4!}{3! \cdot 1!} \right) = 7 \text{ open faults}$$

These 7 different faults correspond to the partitions represented in table 5. This table shows that, even considering only nets broken at a single location, not all the resulting possibilities have to be taken into account. This is due to the fact that all the situations depicted will be detected already if each output is checked to have connection to all inputs. This brings us to the conclusion that it is only necessary to consider those open faults that isolate each output node from the rest of the net (cases (a) and (b) in table 5).

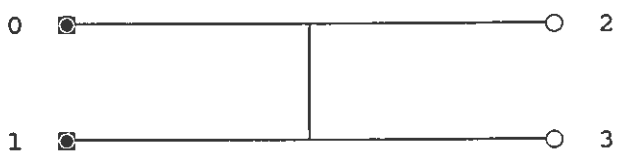
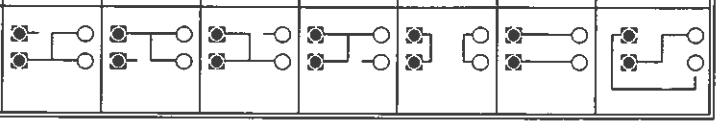


Fig. 3. A simple PCB net.

Table 5. Possible single-location stuck-opens for the net.

output nodes (net 0)	[0]	[1]	-	-	[0,1]	[0]	[0]
input nodes	-	-	[2]	[3]	-	[2]	[3]
output nodes (net1)	[1]	[0]	[0,1]	[0,1]	-	[1]	[1]
input nodes	[2,3]	[2,3]	[3]	[2]	[2,3]	[3]	[2]
visual aid							
	(a)	(b)	(c)	(d)	(e)	(f)	(g)

In general, two conclusions can now be made for testing opens in a net:

1. The number of tests (test patterns) is equal to the number of connected driving outputs.
2. For correct testing, all receiving inputs on the net must be read.

The second conclusion, especially, separates opens from the normal stuck-at faults at inputs. Besides this, there is again no need to consider only one net faulty (broken) at each time, that is, deriving vectors for detection of multiple stuck-open faults can be done considering the occurrence of a stuck-open fault in all nets at once.

6. The Fault Model

For a final conclusion, the fault model to be used for deriving the test methodology in full-BST boards is a mixture of three known models with additional remarks and it will have the following characteristics:

- The three fault types to be considered are:
  - — stuck-at faults;
  - — opens (where it should be specified, if possible, to which value floating inputs are to be updated to);
  - — shorts (where either an or-short, or an and-short, might be specified).
- For stuck-open faults, only those faults that isolate outputs from the remainder of a net are considered, but these faults may occur on multiple nets at once. For the short type of faults, and as far as fault detection is concerned (go/no-go testing of boards), it is enough to consider only 2-net shorts. However, for purposes of fault diagnosis, all shorts between any number of nets should be considered. The diagnostic capability of the test patterns can be enhanced if the type of short is given. When dealing with TTL logic (and-short), the resulting value that detects a short is generally a logic 0. This means that as much 1 as possible should be present in the test patterns checking and-shorts.
- Multiple faults (including more than one fault type) on a certain net will not be considered, since at least one of the faults will always be detected.

6.1. Remarks on Test Pattern Generation

Regarding the structural test approach as a framework from which the actual steps for testing a board are to be derived, and considering the descriptions previously made for building a fault model, it is now time to make some brief comments concerning the important step of test pattern generation (TPG). This will in fact be the basis for a separate paper [16], so this discussion will not be taken very far.

A first remark to be made for the test methodology at TPG level is that any vector inserted in the BST part must obey the following restriction: For any net, and with the exception of open-collector or open-emitter outputs, no more than one driving output may be active at the same time to prevent a short condition on that net. This rule means that the BST control cells corresponding to the enable signals of tri-state outputs, or to the direction control signals of bidirectional pins, should not be initialized in such a way that two or more outputs are active in the same net. This requests a

safe-state insertion process to fill the right positions in a BST serial vector before applying the vector to a circuit. The same remark holds for general outputs that control enable lines of other devices that communicate on the same bus.

A second remark is that the restrictions that follow from the first remark will obviously limit, but not remove the straight use of pseudo-random test-pattern generation schemes, or even of deterministic test-pattern generation schemes (walking sequences, binary counting sequences as in [6] and [7]). This is because the patterns will have to be adapted to the circuit structure, that is, skipping the control cells, and so the application of these techniques becomes less efficient.

The last remark concerns an additional difficulty using BST. The on-board BST test facility is assembled on the same board, with the same processes that made the interconnections. Therefore the test sequence should be preceded by a check on the validity of the BST infrastructure itself, the so called BST *integrity test*. After this first step (checking the BST infrastructure) the board interconnect test can be performed.

6.2. *Application Example of Fault Model and Diagnostics*

In the following, it will be shown that a set of vectors may be able to detect all possible faults (i.e., a 100% fault coverage, within the domain of the actual fault model), but it may happen that complete fault diagnosis is not possible. It is thus of importance to consider this subject here, since it may contribute to validate (or not) the derived fault model.

The example shown in figure 4 will be used to better describe the relationship between detection and diagnosis of faults. This example depicts as a set of 6 nets

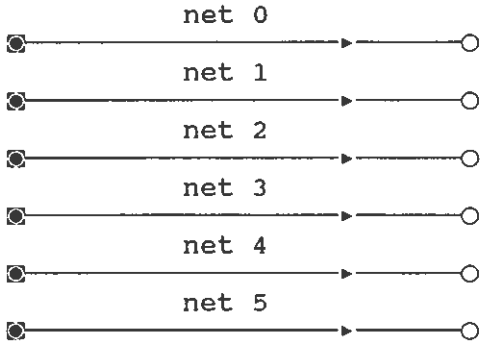


Fig. 4. A set of minimum nets.

of the simplest net topology, but the topology can be generalized to any other set of net types (including tri-state outputs, bidirectional pins, and multiple inputs) by considering that this set represents the state of the circuit at one particular instant of time (one active output, and reading one input at a time).

6.3. *Test Patterns*

Various patterns have been proposed for fault detection using BST, such as in Wagner [6] and Hassan et al. [7]. Some of these patterns result directly from test methodologies much older than BST (Kautz [8]). These proposals result from fault models more or less identical to the one considered here, as can be seen in Wang et al. [4] and Hassan et al. [7].

A simple test pattern that allows detection of all stuck-at faults consists of toggling all nets once (by applying the sequences 11 . . . 1 and 00 . . . 0). A special test pattern that also detects all bridging faults is the binary counting sequence test pattern, which is illustrated in table 6, for the example described. It consists of generating the entier $[\log_2(n + 2)]$ vectors (n represents the number of nets; entier means the first whole number following this result) in such a way that the sequence of values applied to each net will be the equivalent binary value of the decimal number of the net (table 6 shows that this goes from the sequence 001, applied to net 1, up to the sequence 110, applied to net 6). It is obviously necessary to avoid the all-0 and all-1 sequences of values, as these sequences will not allow detection of net stuck-at-1 and net stuck-at-0 faults. For this reason, the number of vectors is given by entier $[\log_2(n + 2)]$ and not entier $[\log_2 n]$. In effect two "fake" nets are introduced to ensure that each net can be vector coded to contain at least one 1 and at least one 0 [17].

It is easy to conclude that the given set of three vectors guarantees a 100% fault coverage, because

- all nets have their value toggled, and thus all stuck-at faults are detected;

Table 6. The binary counting sequence test pattern for the example of figure 3.

NET number	1	2	3	4	5	6
Vector 1	0	0	0	1	1	1
Vector 2	0	1	1	0	0	1
Vector 3	1	0	1	0	1	0

- for the same reason (and since all nets have only one driving output), all stuck-opens are also detected; and
- any shorts that may exist are also detected, since this binary counting sequence has the property of performing a binary search for shorts between all nets (each vector divides all nets in two groups, and checks whether any short exists between nets from these two groups).

6.4. *Diagnosis*

Having seen that a 100% fault coverage is achieved, it is now advisable to take a closer look at fault diagnosis. This section will show that the subject of diagnosis is strongly related with test methodology and test-pattern generation. Considering that in figure 3 a total number of 33 different single faults could take place (12 stuck-ats, 6 stuck-opens, and 15 2-net shorts), we now want to know how precise fault diagnosis can be, based on the result of applying the sequence of 3 vectors shown.

Concerning the stuck-open and stuck-at types of fault, it is obviously possible to pinpoint each faulty net although, in this particular case, because of the simple nets, it is not possible to distinguish between the situations of a net stuck-at and an open net that has its floating inputs updated to the same value as the result of the stuck-at fault.

Table 7 shows the results of applying this sequence of 3 vectors, simulating the 15 possible 2-net short faults. It is considered that the short is of the and-short type. The results on this table show that each simulated fault situation has a unique answer (black underlining). This does not mean that analysis of these results will, in general, allow unique identification of any 2 nets that are shorted together, since some results can also be caused by stuck-at faults (but these could already have been tested for by applying the all-1s and all-0s patterns).

6.5. *Multiple-Fault Diagnosis*

What will happen when more than 2 nets are shorted together? Will the result identify 2 of the nets shorted,

Table 7. Results for the 15 possible 2-net shorts (black lines denote the faulty result to a test vector).

TEST VECTOR on net>	123456	1 SHORT 1-2	2 SHORT 1-3	3 SHORT 1-4	4 SHORT 1-5	5 SHORT 1-6	6 SHORT 2-3
vect. 1	000111	000111	000111	000011	000101	000110	000111
vect. 2	011001	001001	010001	011001	011001	011000	011001
vect. 3	101010	001010	101010	001010	101010	001010	100010

TEST VECTOR on net>	123456	7 SHORT 2-4	8 SHORT 2-5	9 SHORT 2-6	10 SHORT 3-4	11 SHORT 3-5	12 SHORT 3-6
vect. 1	000111	000011	000101	000110	000011	000101	000110
vect. 2	011001	001001	001001	011001	010001	010001	011001
vect. 3	101010	101010	101000	101010	100010	101010	100010

TEST VECTOR on net>	123456	13 SHORT 4-5	14 SHORT 4-6	15 SHORT 5-6			
vect. 1	000111	000111	000111	000111			
vect. 2	011001	011001	011000	011000			
vect. 3	101010	101000	101010	101000			

Table 8. Results for nets (1, 3, 5) and (1, 2, 3, 4) shorted.

TEST VECTOR on net ->	1 2 3 4 5 6	RESULT VECTOR short 135	RESULT VECTOR short 1234
vector 1	0 0 0 1 1 1	000101	000011
vector 2	0 1 1 0 0 1	010001	000001
vector 3	1 0 1 0 1 0	101010	000010

or will it simply be different from any of the results present on table 7? It is not difficult to find two examples that show that both things can happen. Table 8 shows that nets (1, 3, 5) shorted will have the same result as nets (3, 5) shorted, which means that net 1 would not be identified as faulty (it would however be detected after the short (3, 5) was repaired). But the nets (1, 2, 3, 4) are shorted together, the result shows that the circuit is faulty, but it cannot be inferred from table 7 (a single fault dictionary), which is the pair of shorted nets, since it is different from all results present on table 8 (some conclusions can however be drawn, by analyzing the result vectors, with respect to table 7).

Two important conclusions should then be drawn from these comments:

1. For purposes of complete fault diagnosis, it is necessary to include all types of shorts in our fault model, since it may happen that shorts between more than 2 nets are not recognized.
2. The binary sequence test patten will guarantee a 100% fault coverage, but it will not allow precise fault diagnosis, since there are cases possible where faulty nets are masked for diagnosis.

Taking a closer look at the result of applying the sequence of three vectors shown, when nets (1, 3, 5) are shorted, we can conclude that net 1 was masked for diagnosis because:

a. (for vectors 1 and 2) the value on net 1 is already equal to what would result from the (and-) short between nets 3 and 5, which means that the value on net 1 is not influenced because of being involved in the short; and
b. (for vector 3) since the value on net 1 is equal to the values on nets 3 and 5, the value on net 1 does not influence the final result, and thus again remains undetected. Also,
c. if the driving output on net 1 was stuck at 1, in this case with the three-net short, this stuck-at 1 would be pulled down to 0 and left undetected until after repair of the short.

This situation would not happen if we had an or-short instead of an and-short between nets 1, 3, and 5, because then the value on net 1 would influence the values on nets 3 and 5 when vectors 1 and 2 are applied. Covering both types of shorts may however be forced, if each vector and its complement is applied to the circuit, as described in Wagner [6]. Under these circumstances, there will be no more situations where the involvement of a net in a short can be masked for diagnosis, since there will always be a vector for which the value on this net will be changed because of the short. This brings the total number of test patterns to $2 \cdot \text{entier}\,[\log_2(n + 2)]$.

REMARK: The masked stuck-at-1 fault on net 1 still remains unresolved with this double-vector set. If this 3-net short really exists, even the all-1s and all-0s patterns would not detect the stuck-at fault. A stuck-at-1 will be masked by an and-short, a stuck-at-0 by an or-short.

Further studies are still required concerning the adequacy of each specific test-pattern-generation scheme, as far as fault diagnosis is concerned.

7. Conclusions

This article covers the area of fault modeling that is to be reviewed with respect to the new test methodology on board level: Boundary Scan Test. As seen, the fault model is not completely new, but the approach is extended to support detection and diagnosis of faults on PCBs. On the other hand, the introduction of the BST-net implies some reduction of the level of diagnostic accuracy. The following points of interest can be drawn as the major conclusions:

- Because of being "electronic," the BST test facility always needs power. This means that the relatively safe in-circuit unpowered interconnect test is no

longer possible. Serious faults found during initial testing might have to result in an immediate power-down action.

- BST technology is a one-level test solution, applicable to any level but without a description of hierarchical connections.
- Translating open faults and shorts to logical fault models will be technology dependent, because floating inputs will be updated to values that depend on the type of IC technology being used and on fan out and loading conditions that arise on a faulty net.
- Diagnostic capabilities of a set of test vectors can be, based on the applied model, dependent on the technology used on the board under test.
- BST, because of total accessibility, guarantees the possibility of full detectability but not full diagnosis to the level of field replaceable units (FRUs).

Acknowledgments

The authors are cooperating in a European project stimulating information technology: ESPRIT 2478; "Research into Boundary Scan Test Implementation." From this project a partial funding for the investigations is obtained.

The authors wish to thank R.G. Bennetts for his cooperative review.

References

1. JTAG, Final Version 2.0 of the standard Boundary Scan Architecture, April 1988.
2. F. Beenker, K. Eerdewijk, R. Gerritsen, F. Peacock, and M. Star, "Macro testing: Unifying IC and board test," *IEEE Design and Test*, December 1986, pp. 26–32.
3. R. van Riessen, H. Kerkhoff, and A. Kloppenburg, "Design and implementation of a hierarchical testable architecture using the boundary scan standard," *Proc. Ist European Test Conf.*, 1989, pp. 112–118.
4. L. Wang, M. Marhoefer, and E. McCluskey, "A self-test and self-diagnosis architecture for boards using boundary scan," *Proc. Ist European Test Conf.*, 1989, pp. 119–126.
5. Factron-Schlumberger, "The primer of high-performance in-circuit testing," 1985.
6. P. Wagner, "Interconnect testing with boundary scan," *Proc. 18th Inter. Test. Conf.*, 1987, pp. 52–57.
7. A. Hassan, J. Rajski, and V. Agarwal, "Testing and diagnosis of interconnects using boundary scan architecture," *Proc. 19th Inter. Test Conf.*, 1988, pp. 126–137.
8. W. Kautz, "Testing of faults in wiring networks," *IEEE Trans. Comput.*, C-23(4): 358–363, April 1974.
9. J. Hayes, "Fault Modeling," *IEEE Design and Test*, April 1985, pp. 88–95.
10. Texas Instruments, "A primer on testability—test and emulation," 1989.
11. R. Bennetts, "Introduction to digital board testing," Crane Russak & Company, Inc., 1982 (ISBN 0-8449-1385-0).
12. H. Fujiwara, "On the acceleration of test generation algorithms," *IEEE Trans. Comput.*, C-32(12): 1137–1144, 1983.
13. T. Kirkland, and M. Mercer, "Algorithms for automatic test pattern generation," *IEEE Design and Test*, June 1988, pp. 43–55.
14. S. Vining, "Tradeoff decisions made for a P1149.1 controller design," *Proc. 20th Inter. Test Conf.*, 1989, pp. 47–54.
15. F. Francese, "Il collaudo di circuiti su piastra stampata," *Alta Frequenza*, 57(6): 177–187, 1988.
16. F. de Jong, "Test pattern generation for boards with partial Boundary Scan Test." (Paper under preparation.)
17. P. Goel and M.T. McMahon, "Electronic chip-in-place test," *Proc. 13th Inter. Test Conf.*, 1982, pp. 83–90.
18. IEEE/P1149.1 Working Group, IEEE Std. 1149.1-1990 Standard Test Access Port and Boundary-Scan Architecture, February 1990.

Frans de Jong is a member of the Centre for Manufacturing, a central R&D and support division for manufacturing within Philips (Eindhoven, The Netherlands). He received an M.Sc. in Electronics from the Delft University (Netherlands) in 1985. At Philips he has done R&D for PCB testing, mixed-signal testing, and DSP activities in relation to board testers. Last year he worked on digital testing and the application of Boundary Scan Test.

José S. Matos holds an M.Sc. and a Ph.D. degree in Electrical Engineering from Syracuse Univerity, Syracuse, New York. He leads a group working in CAD and Micro electronics at INESC (Instituto de Engenharia de sistemas e Computadores) in Porto, Portugal; and he is an Assistant Professor at the Faculty of Engineering of the University of Porto. His interests include synthesis, VLSI education and test.

José M. Ferreira obtained hs M.Sc. degree from the University of Porto, Portugal in 1987. He is currently with INESC, where he works in digital systems design and test. He also lectures at the Faculty of Engineering of the Univerity of Porto, where he is working toward a Ph.D. degree.