J Electron Test (2009) 25:187–195
DOI 10.1007/s10836-008-5091-1

A New Algorithm for the Selection of Control Cells in Boundary-Scan Interconnect Test

A. Quiros-Olozabal · M. A. Cifredo-Chacon

Received: 18 December 2007 / Accepted: 22 September 2008 / Published online: 30 January 2009

Abstract This paper presents an algorithm for the generation of the values to be loaded in the control cells of a Boundary-Scan (BS) chain during an interconnect test. The algorithm selects several groups of control cells while avoiding that two or more drivers excite the same net at the same time, allowing every net to be active for every test vector and testing every driver after the execution of the overall test process. It allows for 100% detection of short, open, stuck-at and driver transition faults on fully controllable and observable BS nets on virtually any BS board. In fact, only two minor requirements are imposed: (1) the sets of nets affected by two different control cells must be disjoint or one of them must be included in the other; (2) every net of a set affected by a control cell must have the same number of drivers. In addition, the algorithm can be implemented very easily, avoiding the need to explore all the possible combinations of values to be loaded in the control cells.

Keywords Boundary-scan · Interconnect test · Control cell selection

1 Introduction

Interconnect testing of digital electronic systems has to take into account the possibility that a significant number of nets can have several drivers. This fact implies not only deciding which values must be applied to each net, but also which drivers are going to be used to apply those values.

Responsible Editor: E. J. Marinissen

A. Quiros-Olozabal (✉) · M. A. Cifredo-Chacon
Microelectronic Circuits Design Group, University of Cadiz,
Escuela Sup. de Ingenieria, C/Chile, 1-11003 Cadiz, Spain
e-mail: angel.quiros@uca.es

Many contributions have been made concerning algorithms used for the generation of values to be sent to the nets. Some of them, such as the counting sequence [10], the modified counting sequence [6], the walking one sequence or the maximal independent set [14] are well-known and have been exhaustively analyzed [9, 14], while more recent contributions, such as the group, net, shifted net (GNS) sequence [11] continue exploring the possibilities of combining the best aspects of those well-known algorithms: high detection and diagnosis capability and short testing periods.

All of these algorithms assume that the stimulus can be applied to every net without restrictions, but previous work must be carried out in order to guarantee this capability.

First, and in order to avoid electrical problems (known as *driver contention*), only one driver must to be enabled for each net and test vector. The rest of the drivers for that net must be disabled.

Second, in order to maintain net fault coverage as high as possible, the maximum number of nets has to be stimulated in every test vector.

Third, in order to reach a high driver fault coverage, the maximum number of drivers must be used during the overall test process.

Finally, these three conditions should be fulfilled with the minimum number of test vectors, in order to achieve a short test period.

The adoption of the IEEE 1149.1 standard [8] allows for the systematic selection of the drivers to be used. In a BS component, a control cell is assigned to every output cell that can be connected to others outputs in the same net. The selection of the drivers to be used in each test vector is accomplished by loading the control cells of every component on a board with the *correct values*.

The standard's flexibility, however, makes this selection an important task because an output cell may be controlled by another cell or by itself (i.e. an open-collector output), and a control cell can control one or several output cells. Since the decision of adopting the control structure is made by the component's manufacturer, it is possible that the same group of nets on a BS board (i.e. a data bus) has only one control cell in a component, and several control cells in another one (Fig. 1).

At the same time, a group of controlled nets can be connected to a single component on one part of the board (i.e. a data bus connected to a microprocessor) and can be divided in two or more components on another section of the board (i.e. the same data bus connected to two memory IC's, Fig. 1).

Finally, there are no restrictions in the standard concerning the order of the cells inside a BS component, or the order of the components in the BS chain.

Several solutions for the selection of drivers in BS interconnect test have been proposed. The simplest one consists of activating only one control cell at a time, using a walking sequence for the generation of the values to be loaded in the control cells. This method can be typically found in Built-In Self Test (BIST) approaches [2, 13], where an easy to implement solution is a key aspect. Adopting this solution, driver contention is avoided but, from all the nets of the board that have multiple drivers, only those nets controlled by the activated control cell are driven. So, in order to detect all the possible short faults, non-driven nets must behave as they have a logical one value or a logical zero value, and this is not true in general for tristate nets. In addition, long test times will be required to test all possible driver faults because a specific partial test is needed for each control cell.

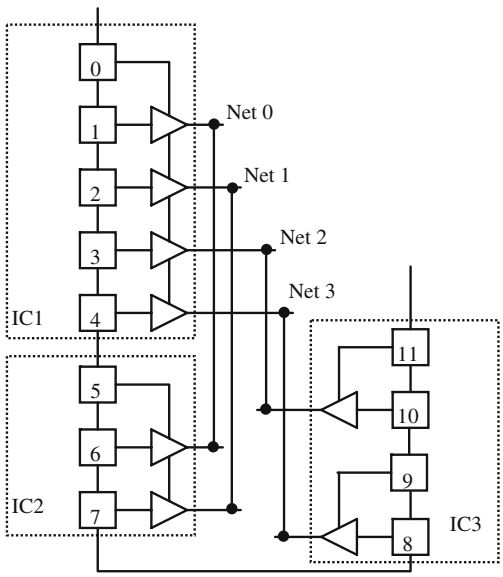

Fig. 1 Different control structures for a bus

More comprehensive BIST [3, 12] and non-BIST approaches [1, 4, 5] propose the use of groups of simultaneously activated control cells to reduce the number of test vectors and to increase the net fault coverage. These groups of activated control cells must be selected by complying with the following conditions:

1. Two or more drivers for the same net are never simultaneously enabled.
2. Every net is active in every group of enabled control cells.
3. Every driver is used by at least one of the groups of enabled control cells.

In this way, the conditions to avoid driver contention and to reach high fault coverage are fulfilled. At the same time, the number of test vectors needed to test all possible driver faults is reduced because several drivers can be simultaneously enabled and tested.

Unfortunately, these authors do not include an algorithm to be used for the selection of groups of activated control cells, therefore, fulfilling the previously cited conditions. Consequently, the proposed solutions are not complete because this is not a trivial task.

Only in [4] it could be considered that the selection of these groups of control cells is an easy job because it is assumed that every driver has its own and exclusive control cell, but this assumption is not true in general.

The selection of the control cells to be activated can be performed by an algorithm that is based on the analysis of all possible combinations of activated control cells using binary decision diagrams, such as the one in [7]. Since there may be many combinations ($2^{Nccells}$, where N_{ccells} is the number of control cells), several grouping and sorting processes are performed, and the netlist is previously partitioned, including the potentially conflictive control cells in each partition. As such, the magnitude of the problem can be reduced, but the resulting algorithm is quite complex and requires extensive analysis and preliminary work.

In this paper, we propose an algorithm that is much easier to implement and can be used to find a solution while avoiding the need to explore all the possible combinations of activated control cells. Consequently, its use is as suitable for BIST as for non-BIST approaches. With minimal preliminary work, the process basically consists of activating the first unused control cell for each output cell found in the BS chain. By repeating this process a number of times, equal to the maximum number of joined drivers, the previously described groups of activated control cells can be obtained.

The assumed net and fault models are presented in Sections 2 and 3, respectively. The input and output information is described in Section 4. Section 5 includes the generation algorithm and an example of application. The algorithm is formally analyzed in Section 6, and an overview of its use in a BS interconnect test is the contents of Section 7. Finally, experimental results and conclusions are presented in Sections 8 and 9, respectively.

2 Net Model

The control cell selection process described here takes any net connected to the BS chain into account, although the highest failure coverage of the overall test process is guaranteed only for those nets in the board that are fully controllable and observable from the BS chain. A net is fully BS controllable and observable if these three conditions are fulfilled:

1. There is at least one output or bi-directional cell connected to the net.
2. There is at least another input or bi-directional cell connected to the net.
3. The net can be assigned any value from the BS chain without conflicting with any other of the board's non-BS components.

The first two conditions guarantee that the net can be assigned a value from the BS chain and that the value of the net can be read from the BS chain. A single bi-directional cell connected to a net is not sufficient because an open failure between the pin and the net would not be detected.

The third condition expresses the need to have the capability of controlling a net without causing an electrical problem.

One or several input and output cells may be connected to each net. The *degree* of a net is the number of output cells connected to it.

A net is *simple* if only one output cell is connected to it and this cell cannot be disabled. A net is *wire-and* (*wire-or*) if one or more output cells are connected to it and the value of the net is the logical AND (OR) function of the values of these cells. In a net of this kind, a disabled output is equivalent to a weak logical one (zero) value, and this is the value of the net when all those outputs are disabled. Typically, this weak value is obtained by the use of a pull-up (pull-down) resistor. A net is *tristate* if one or more output cells are connected to it and these cells can be disabled to a high impedance state.

It is assumed that every BS cell is connected to one net, at most. The IEEE 1149.1 standard allows a cell to be connected to two nets if an input (or bidirectional) pin is used exclusively as a source of an output (or bidirectional) pin, with null system logic[1] between them [8]. For this situation, the cell is treated as an output by the proposed algorithm and the net considered is the one connected to the output pin.

A free control configuration is assumed:

1. More than one output cell can share a single control cell.
2. There are no restrictions concerning the order of the cells in the BS chain.
3. Control cells may be active low or active high.

In order to guarantee that the highest fault coverage can be reached, only two conditions are imposed as to the structure of buses and control cells:

1. The sets of nets affected by control cells i and j must be disjoint or one of them must be included in the other.
2. Every net of a set affected by a control cell must have the same degree.

These conditions are easily fulfilled if recommendation 11.6.1 (q) of IEEE standard 1149.1 is followed [8] ("The control signal for each functionally distinct group of system output pins should be driven from a distinct boundary-scan cell dedicated to that purpose, even where the output from the on-chip system logic observed by that cell normally would drive a common control signal to more than one such group of pins".)

If one or both of these conditions is not fulfilled, the proposed algorithm still works, but the fault coverage could be lower than 100% for the short faults.

The imposed conditions can be assumed in the vast majority of practical situations, and they are compatible with any control structure: a different control cell for every output cell, a bus with the same control cell for all the nets that form it, a bus controlled by two or more control cells and different control structures for the same bus in different components.

3 Fault Model

The objective of the BS interconnect test is to detect three types of defects:

1. Shorts.
2. Opens.
3. Defects in the input or output cells.

[0] Devices such as buffers are not considered to be "logic".

These defects can produce a variety of faults that are summarised in Table 1.

This large fault set can be simplified if it is taken into account that the capability to detect some of these faults implies the capability to detect others.

First, the capability to detect any stuck-at fault at the inputs implies the capability to detect any stuck-at fault at nets.

At the same time, the capability to detect any stuck-at or transition fault at the outputs implies the capability to detect any stuck-at or transition fault at the inputs, simply because the behaviour of the outputs is *read* through the inputs.

This capability to detect faults at output and input cells also implies the capability to detect opens, because the open will affect the behaviour of one or several outputs and/or inputs.

Finally, a permanently disabled output is the same as an open between the output and the net, so this fault does not need specific consideration.

As such, the set of faults to be taken into consideration can be minimised and is summarised in Table 2.

Considering the problem of the selection of control cells, the detection of 100% of short faults implies the use of a group of control cells that simultaneously activates all the nets in the board. This is because non-driven tristate nets do not generally behave in a predictable way, so a short fault in a non-driven tristate net would not be detected.

At the same time, the detection of the faults related to output cells implies enabling every output cell at least during part of the test process. The generation algorithm must then select several groups of control cells avoiding driver contention, activating all nets and using every driver in at least one of these groups.

Table 1 The interconnect test aims to detect the following defects

Defect	Fault
Short net	Dominant net
	AND type short
	OR type short
	Stuck at 0 net
	Stuck at 1 net
Open net	Stuck at 0 input
	Stuck at 1 input
Input cell	Stuck at 0 input
	Stuck at 1 input
	Transition fault
Output cell	Stuck at 0 output
	Stuck at 1 output
	Transition fault
	Always enabled output
	Always disabled output

Table 2 Minimum set of faults to be taken into consideration

Defect	Fault
Short net	Dominant net
	AND type short
	OR type short
Output cell	Stuck at 0 output
	Stuck at 1 output
	Transition fault
	Always enabled output

4 Input and Output Information

The generation algorithm needs the information contained in the netlist and BSDL (Boundary-Scan Description Language) files. In order to simplify the generation process, however, three arrays are built-up extracting part of that information. These arrays provide the input for the generation algorithm and their length is equal to the total number of cells, which is simply the sum of the boundary lengths of every component in the chain. These lengths are extracted from the BOUNDARY_LENGTH attribute in each BSDL file.

If the total length is represented by *BRL* (Boundary Register Length), the elements of these arrays are numbered from *BRL-1* to *0*, following the order of components in the board and the order of cells inside each component, always from TDI to TDO.

These three arrays contain the following information for each cell in the boundary-scan chain respectively:

1. The type of cell. For the generation process, it must be known if the cell acts as an output that can be disabled or not. Only the bidirectional or output cells that are non-simple (the ones that can be connected to other outputs) must be taken into account. Every cell that has the ccell element of the BOUNDARY_REGISTER attribute of any value in its BSDL file will be considered a non-simple output cell.

In the example circuit of Fig. 2 all output and bidirectional cells will be considered as non-simple outputs because all of them are associated to a control cell.

2. The position of its control cell. The value of the ccell element indicates the position of the control cell in the BS chain, but the information contained in each BSDL file refers each component's boundary register, so it must be corrected and refer to the total length of the boundary chain.

Again for the circuit of Fig. 2, for bidirectional cells 1 to 4 the position of their control cell is 0; for output cells 6 and 7 the position of their control cell is 5, etc.

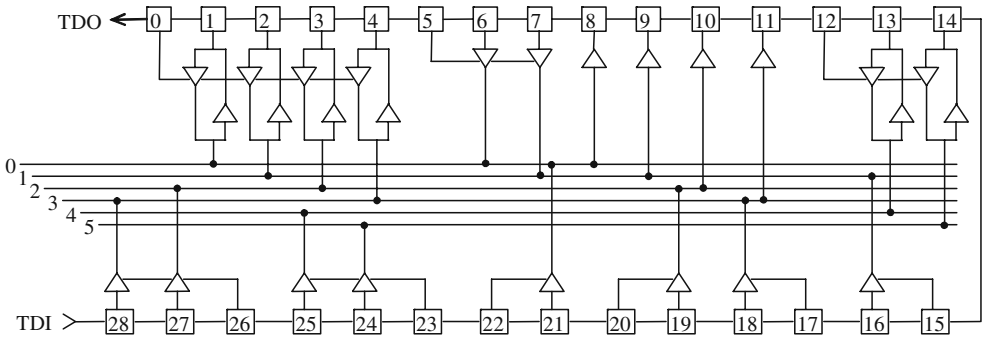

Fig. 2 Example circuit

3. The net that the cell is connected to. This information is extracted from the netlist and BSDL files. A different number is assigned to every net.

In the example circuit cells 1, 6 and 8 are connected to net 0; cells 2, 7, 9 and 16 are connected to net 1, etc.

The complete input arrays for the circuit in Fig. 2 are shown in Table 3. This example circuit has 6 nets, 29 cells and its maximum degree is 3.

Using the algorithm described in the following section, an array of binary values is obtained. This array has a number of rows equal to the number of groups of activated control cells, which has to be equal to the maximum degree of the nets on the board (3 in the example circuit), and has a number of columns equal to the number of cells (29 in the example circuit). The corresponding element indicates with a logical value 1 (0) that the control cell will be activated (deactivated) for a given group and control cell. The contents of the corresponding column in this array are not significant for non-control cells.

This output array must be processed by taking into account the value of the safe element of the BOUNDARY_REGISTER attribute for each control cell. This element indicates the logical value that deactivates each control cell.

5 Generation Process

The first step of the generation process builds up two intermediate arrays. The first one, with a number of columns equal to the number of cells, and a number of rows equal to the number of nets, summarizes the correspondence between nets and control cells. The second array, with one column and a number of rows equal to the number of nets, contains the degree of every net involved in the generation process.

This first step analyzes the boundary register and, for every cell that is connected to a non-simple output, marks the element of the first intermediate array corresponding to the net and control cell related to that output cell.

When the first array has been completed, the number of marked elements for each net (row of the first intermediate array) is counted, obtaining the degree of the net that is stored in the corresponding row of the second intermediate array. The pseudo code for this step can be found in Fig. 3.

The resulting intermediate arrays in the example circuit are shown in Table 4.

The second step of the generation process uses the intermediate arrays obtained after the first step and two auxiliary arrays that indicate if each control cell and net respectively has already been processed during the execution of the algorithm. Both auxiliary arrays have a number of rows equal to the number of groups of control cells, and the number of columns is equal to the number of cells and to the number of nets, respectively.

This second step begins initializing the output array with all control cells deactivated for every group of control cells. Similarly, the auxiliary arrays are initialized with all cells and nets as not processed for every group of control cells.

Next, a loop begins with a number of iterations equal to the number of groups of control cells (equal to the maximum degree of the board). For each iteration, a group

Table 3 Input arrays for the example circuit

Cell acts as a non-simple outputs																												
28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	1	0	0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	1	1	0	1	1	1	1	0
Position of the control cell																												
28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
26	26	–	23	23	–	–	22	–	20	17	–	15	–	12	12	–	–	–	–	–	5	5	–	0	0	0	0	–
Net that the cell is connected to																												
28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
3	2	–	4	5	–	–	0	–	2	3	–	1	–	5	4	–	3	2	1	0	1	0	–	3	2	1	0	–

```
Initialize net-vs-control_cell array "all not marked"
For each cell loop
 If cell is not simple output then
   Mark the corresponding net (row) and control cell
   (column) of net-vs-control_cell array
 End if
End loop
For each net loop
 Initialize degree as zero
 For each cell loop
   If element of the net-vs-control_cell array is
   marked then
    Increase degree
  End if
 End loop
 Store degree in the element of degree array
End loop
```

Fig. 3 Pseudo code for the first step

of control cells is selected for activation. Every time a control cell is activated, all the nets controlled by this cell and all the control cells that are related to those nets are marked as processed in the corresponding row. In this way, the selection process can take into account the control cells and nets that are previously processed by the activation of another control cell.

Each iteration has two inner loops with the same number of iterations and cells. In the first one, the nets considered are the ones that have a degree higher or equal to the number of the group of control cells that it is been selected. Not all the control cells have been used in these nets, so it must be taken into account whether the processed control cell has been previously activated in another group. In this case, the control cell has to be skipped in order to allow another control cell to participate in the test process.

When this first inner loop has finished, the degree of the remaining nets is lower than the number of the group. All the control cells have been used in a previous group for these nets, so the first control cell found in the BS chain that can be enabled without generating an electrical problem is selected. The pseudo code for this step can be found in Fig. 4.

In the example circuit, the number of groups of control cells required is three.

For the first group, all the nets have a degree higher than the number of the group (1), so all the nets receive the same treatment and the second inner loop is inactive. The control cells that are activated for this first group (26, 23, 22 and 15) are the first ones found in the BS chain for each net that do not generate an electrical conflict with a previously enabled control cell.

In the second outermost iteration, the second group of control cells is selected. Again all nets have a degree higher or equal to the number of the group (2), so all nets receive again the same treatment. But in this case, the result of the first iteration is taken into account and the previously activated control cells are skipped. So, control cells 26, 23, 22 and 15 are not activated in order to allow the activation of control cells 20, 17, 12 and 5.

The third (and last) outermost iteration does not apply the same treatment to all nets because nets 0, 1, 2 and 3 have a degree equal to the number of the group (3), but nets 4 and 5 have a degree (2) lower than the number of the group. So in the first inner iteration for the cells, nets 0, 1, 2 and 3 are taken into account, and the control cells previously used for these nets (26, 22, 20, 17, 15 and 5) are skipped, enabling activation of control cell 0.

On the other hand, nets 4 and 5 are considered by the second inner iteration, which does not take into account if the control cell has been previously used, so control cell 23 is enabled again, as in the first group. The output and the auxiliary arrays after the second (and final) inner iteration are included in Table 5.

Table 4 Intermediate arrays for the example circuit

Net vs control cell array

	...	26	...	23	22	...	20	...	17	...	15	...	12	...	5	...	0
0					X										X		X
1											X				X		X
2		X					X										X
3		X							X								X
4				X									X				
5				X									X				

Degree array

0	3
1	3
2	3
3	3
4	2
5	2

```
Initialize output array as "all disabled"
Initialize auxiliary arrays as "not processed"
For each group of control cells loop
 For each cell loop
  If cell is not simple output and
   net degree is not lower than group and
   net is not processed and
   control cell is not processed and
   control cell has not been previously used then

   Activate control cell in output array in this group
   For every net controlled by the control cell loop
    Mark the net as processed for this group
    For every control cell related to the net loop
     Mark the control cell as processed for this group
    End loop
   End loop

  End if
 End loop
 For each cell loop
  If cell is not simple output and
   net is not processed and
   control cell is not processed then

   Activate control cell in output array in this group
   For every net controlled by the control cell loop
    Mark the net as processed for this group
    For every control cell related to the net loop
     Mark the control cell as processed for this group
    End loop
   End loop

  End if
 End loop
End loop
```

Fig. 4 Pseudo code for the second step

6 Formal Analysis

This section's objective is to prove that the proposed algorithm generates the groups of active control cells complying with the conditions included in Section 1, and will show how the restrictions imposed to the BS circuits in Section 2 affect the generation process.

First, driver contention must be avoided with no exceptions. It can be assured that the proposed algorithm complies with this condition because every time a control cell is activated in a group, all the remaining control cells that are related to every net controlled by the active control cell are marked as processed and, consequently, deactivated in that group. With the proposed algorithm, there are no restrictions imposed to the BS circuit to fulfill this condition.

Second, every net must be driven in every group of active control cells. Imposing the restrictions of Section 2, and considering fully BS observable and controllable nets, it can be proven that the proposed algorithm complies with this condition.

If a net is not driven, then all its control cells are deactivated. If a control cell (i) is deactivated, then there is at least another net controlled by it that is driven from an active control cell (j). If the condition 1 imposed to the BS structure in Section 2 is fulfilled, then the set of nets controlled by one of the control cells is included in the set of nets controlled by the other control cell. But if the set of nets controlled by control cell i is included in the set of nets controlled by control cell j, then the non-driven net would be driven because control cell j is active. On the other hand, if the set of nets controlled by control cell j is included in the set of nets controlled by control cell i, then there must be another control cell for the non-driven net that can be activated, because the condition 2 imposed on the BS structure in Section 2 indicates that every net affected by a control cell must have the same degree. So, if conditions 1 and 2 imposed on the BS structure in Section 2 are fulfilled, then all nets are driven in every group of active control cells.

If either or both of the conditions imposed in Section 2 are not fulfilled, some of the nets may be non-driven in some of the groups of active control cells. This does not, however, due to the algorithm itself, because these nets would be non-driven in order avoid driver contention.

Table 5 Output and auxiliary arrays for the example circuit

Output array

	...	26	...	23	22	...	20	...	17	...	15	...	12	...	5	...	0
1		1		1	1		0		0		1		0		0		0
2		0		0	0		1		1		0		1		1		0
3		0		1	0		0		0		0		0		0		1

"Net is processed" array

	0	1	2	3	4	5
1	1	1	1	1	1	1
2	1	1	1	1	1	1
3	1	1	1	1	1	1

"Cell is processed" array

	...	26	...	23	22	...	20	...	17	...	15	...	12	...	5	...	0
1		1		1	1		1		1		1		1		1		1
2		1		1	1		1		1		1		1		1		1
3		1		1	1		1		1		1		1		1		1

Table 6 Algorithm's run-time and number of test vectors

Circuit	BR Length	Nets	Maximum degree	Run-time (ms)	Test vectors
1	36	16	1	0.2	6
2	160	36	3	2.7	12
3	196	52	3	4.7	12
4	190	66	3	5.0	13
5	226	82	3	7.1	13
6	430	100	4	15.8	15
7	422	134	3	22.6	14
8	582	130	5	27.4	18
9	620	166	4	38.3	16
10	772	196	5	56.5	18
11	860	200	4	60.3	16
12	1,012	230	5	82.5	18

When this situation appears, it is interesting to know which group of active control cells drives the maximum number of nets, because this group is the most adequate to be used in the interconnect test phase that detects the faults at nets. This group can be simply determined by counting the number of processed nets in the auxiliary array at the end of the generation process.

Finally, all the control cells must be included in at least one of the groups of active control cells. It is easy to prove that this condition is fulfilled because while the degree of a net is equal or lower than the group that is being processed, the previously activated control cells for that net are skipped, so every control cell is activated sooner or later by the generation process.

7 Algorithm Use in a BS Interconnect Test Process

The overall interconnect test process will be defined not only by the groups of selected drivers obtained by the proposed algorithm, but also by the values sent to the nets from the selected drivers. Consequently, fault coverage and test length will depend on both aspects of the test process.

It can be assured, however, that the proposed algorithm allows for 100% fault detection in the previously cited conditions because all nets participate in every test vector and every driver is used during overall test process. At the same time, this algorithm allows that a minimum number of test vectors be used because the number of different groups is as low as possible (equal to the maximum degree of the nets of the board).

A BS interconnect test process is typically divided in two phases: a test for shorts and a test for opens, stuck-at faults and driver faults. In the first phase, a counting sequence provides 100% detection capability for shorts with a minimum number of test vectors given by expression 1, where N_{tvec1} is the number of test vectors and N_{nets} is the number of nets.

$$N_{tvec1} = [\log_2(N_{nets})] \quad (1)$$

Any of the groups of drivers obtained by the proposed algorithm can be used for this first phase because any of them activates all the nets of the board. For this group of drivers many of the opens, stuck-at faults and driver faults can also be detected during this phase, but to reach 100% fault coverage some additional test vectors would be needed. Consequently, this group of drivers receives the same treatment in the second test phase.

In this second phase two test vectors are applied to the nets from each group of drivers. These two vectors are the logical one and zero values, and they are obtained inverting successively the values sent to the nets in the last vector of the first phase. In this way the logical value one and the logical value zero and the two possible transitions (from 0 to 1 and from 1 to 0) are sent to every net from each one of its drivers. The number of vectors for the second phase is given by expression 2, where N_{tvec2} is the number of test vectors and D_{max} is the maximum degree of the nets of the board.

$$N_{tvec2} = 2 \times D_{max} \quad (2)$$

The total number of test vectors is simply the sum of N_{tvec1} and N_{tvec2}.

8 Experimental Results

The presented algorithm has been implemented in Visual Studio.Net 2003 (Visual Basic language) and tested with several BS example circuits to corroborate the results of the formal analysis and to evaluate the algorithm's run-time.

The results have been obtained in a conventional desktop computer (1.6 GHz, 1 Gbyte of RAM) under Microsoft Windows XP Professional Edition.

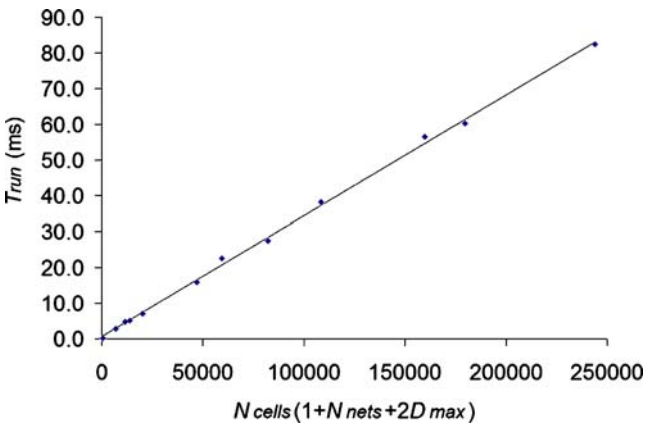

Fig. 5 Algorithm's run time vs circuit complexity

The characteristics of the circuits, the algorithm's run-times and the resulting number of test vectors using the approach of the previous section are shown in Table 6.

By analyzing the algorithm, it can be ascertained that the run-time should be dependent on the circuit parameters shown in expression 3.

$$T_{\text{run}} \propto N_{\text{cells}}(1 + N_{\text{nets}} + 2D_{\max}) \quad (3)$$

Where T_{run} is the run-time, and N_{cells} is the number of cells. If the proportionality constants of expression 3 are evaluated for the example circuits, the standard deviation of those constants is lower than a 10% of its average value, showing that expression 3 fits quite well with the results in Table 6. This linear dependence is shown in Fig. 5.

Expression 3 can be approximated by expression 4 when the number of nets exceeds the maximum degree of the nets of the board.

$$T_{\text{run}} \propto N_{\text{cells}} \times N_{\text{nets}} \quad (4)$$

This formula can be used to estimate the proposed algorithm's run-time for large circuits.

9 Conclusion

The correct selection of the control cells to be activated during BS interconnect test is important in order to obtain high fault coverage and short testing time. We have presented a new algorithm that can select several groups of active control cells avoiding driver contention, allowing every net to be driven for every test vector, and testing every driver after the execution of the overall test process. In this way, a high fault detection capability can be obtained.

Only two conditions are imposed on the circuit in order to obtain the maximum fault detection capability and, if these conditions are not fulfilled, the algorithm still works well, but some nets can be inactive during part of the test procedure in order to avoid electrical problems.

The exhaustive analysis of all the possible combinations of enabled control cells is not needed and the algorithm is quite simple and easy to implement.

The run-time for the selection process is short and approximately proportional to the product of the number of cells times the number of nets.

References

1. Angelotti, F. W., et al. (1993). System level interconnection test in a tristate environment. Proceedings IEEE International Test Conference, pp. 45–53.
2. Chiang, C., & Gupta, S. K. (1997). BIST TPGs for faults in board level interconnect via boundary scan. Proceedings IEEE VLSI Test Symposium, pp. 376–382.
3. Feng, W., Huang, W. K., Meyer, F. J., & Lombardi, F. (1999). A BIST TPG approach for interconnect testing with the IEEE 1149.1 STD. Proceedings Asian Test Symposium, pp. 95–100.
4. Feng W., Karimi F., Lombardi F. (2001) Fault detection in a tristate system environment. *IEEE Micro 21*(5):77–85.
5. Feng, W., Meyer, F. J., & Lombardi, F. (1999). Novel control pattern generators for interconnect testing with boundary scan. Proceedings International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 112–120.
6. Goel, P., & McMahon, M. T. (1982). Electronic chip-in-place test. Proceedings International Test Conference, pp. 83–90.
7. Her, W., Jin, L., & El-Ziq, Y. (1992). An ATPG driver selection algorithm for interconnect test with boundary scan. Proceedings International Test Conference, pp. 382–388.
8. IEEE.Std. 1149.1-2001 (Revision of IEEE Std. 1149.1-1990) IEEE Standard Test Access Port and Boundary-Scan Architecture., The Institute of Electrical and Electronic Engineers, Inc., 2001.
9. Jarwala, N., & Yau, C. W. (1989). A new framework for analyzing test generation and diagnosis algorithms for wiring interconnects. Proceedings International Test Conference, pp. 63–70.
10. Kautz W. H. (1974) Testing for faults in wiring interconnects. *IEEE Trans Comput c-23*(4):358–363 Apr.
11. Kim Y., Kim H., Kang S. (2004) A new maximal diagnosis algorithm for interconnect test. IEEE Transactions on Very Large Scale Integration (VLSI). *Systems 12*(5):532–537
12. Su, C., & Chen, Y. (1998). Comprehensive interconnect BIST methodology for virtual socket interface. Proceedings Asian Test Symposium, pp. 259–263.
13. Su C., Jou S. (1999) Decentralized BIST methodology for system level interconnects. *J Electron Test* 15(3):225–265 Dec.
14. Yau, C. W., & Jarwala, N. (1989). A unified theory for designing optimal test generation and diagnosis algorithms for board interconnects. Proceedings International Test Conference, pp. 71–77.

Angel Quiros-Olozabal received a B.Sc. degree in Electronic Engineering from the University of Cadiz, Cadiz, Spain, in 1987, a B.Sc. degree in Physics, specialized in Electronics, from UNED, Madrid, Spain, in 1991, and a Ph.D. in Industrial Electronics from the University of Cadiz, Cadiz, Spain, in 2002. He has been an associated lecturer from 1987 to 1996 and a titular lecturer since 1996, in the Department of Systems Engineering and Electronics at the University of Cadiz. His research is focused on Boundary-Scan test and synthesis of electronic circuits from VHDL descriptions.

Maria Angeles Cifredo-Chacon received a B.Sc. degree in Electronic Engineering, in 1994 and a B.Sc. degree in Industrial Organization Engineering from the University of Cadiz, Cadiz, Spain, in 1997. She has been an associated lecturer from 1998 until present, in the Department of Systems Engineering and Electronics at the University of Cadiz. Her research is focused on synthesis of electronics circuits from HDL descriptions.