ATPG for Dynamic Burn-In Test in Full-Scan Circuits

ATPG FOR DYNAMIC BURN-IN TEST IN FULL-SCAN CIRCUITS

Alfredo BENSO, Alberto BOSIO, Stefano DI CARLO, Giorgio DI NATALE, Paolo PRINETTO
Politecnico di Torino, Dipartimento di Automatica e Informatica
Corso Duca degli Abruzzi 24, I-10129, Torino, Italy
E-mail: {alfredo.benso, alberto.bosio, stefano.dicarlo, giorgio.dinatale, paolo.prinetto}@polito.it

Abstract

Yield and reliability are two key factors affecting costs and profits in the semiconductor industry. Stress testing is a technique based on the application of higher than usual levels of stress to speed up the deterioration of electronic devices and increase yield and reliability. One of the standard industrial approaches for stress testing is high temperature burn-in. This work proposes a full-scan circuit ATPG for dynamic burn-in. The goal of the proposed ATPG approach is to generate test patterns able to force transitions into each node of a full scan circuit to guarantee a uniform distribution of the stress during the dynamic burn-in test.

1. Introduction

Yield and reliability are two key factors affecting costs and profits in the semiconductor industry [1] [2].

There are a high number of physical failure mechanisms that can affect the reliability of an electronic component. Among them, common ones include TDDB (Time Dependent Dielectric Breakdown), hot carrier aging, and electro migration. In addition, certain fabrication steps can cause stresses that may lead to latent damages with a consequent reduction of the device lifetime.

Stress testing is a technique based on the application of higher than usual levels of stress to speed up the deterioration of electronic devices. The idea behind this screening process is to accelerate the lifetime of a device and let it begin its normal operation with a failure rate out of the so-called infant mortality region.

The standard industrial approaches for stress testing are high temperature burn-in [3] and high-voltage screening [4].

Burn-in has been demonstrated to be effective in varying degrees for almost all circuits and assembly causes of permanent failure. In particular burn-in screening is able to decrease the failure rate of a product during the early field life, where overall cost and turn around time are of concerns. The added manufacturing cost may range from 5% to 40% of the total product cost, depending on the burn-in time, qualities of ICs, and product complexity [5].

Careful attention to the design of stresses for burn-in is necessary to ensure that latent defects are accurately discovered and on the other side that the useful life of remaining devices is not adversely affected.

In particular, in the case of so-called dynamic burn-in (see Section 2) the design of test patterns able to force transitions in a uniform way in all the part of a circuit under test is an open issue.

This work proposes a full-scan circuit ATPG for dynamic burn-in. The goal of the proposed ATPG is to generate test patterns able to force transitions into each node of a full scan circuit to guarantee a uniform distribution of the stress during the dynamic burn-in test.

In addition, our approach attempts to equalize the transitions forced into the circuit in order to avoid overstressing part of the device and possibly damaging it.

The paper is organized as follows: Section 2 introduces the burn-in and the problem of generating test patterns for dynamic burn-in; Section 3 overviews the proposed ATPG approach, and Section 4 shows some experimental results obtained by applying the ATPG on a set of benchmarks. Finally, Section 5 draws some conclusions.

2. The problem of the burn-in test

Burn-in is a screening method used to eliminate defective components in completely processed microelectronic parts utilizing accelerated aging. This screening is obtained by applying stressful operation conditions.

Mainly we can distinguish three types of burn-in [6][7]:

- The *simple state burn-in* consists in applying bias and temperature and performing component test before and after burn-in. The

15th Asian Test Symposium (ATS'06)
0-7695-2628-4/06 $20.00 © 2006 IEEE

input and output pins are open and only the power supply pins are connected. This simple board design allows stressing many parts in parallel at a low cost. The drawback is that not all the nodes of the circuit are excited and there is no feedback during burn-in, which results in the highest escape rate of all burn-in types;

- The *dynamic burn-in* addresses the potential problem of not all nodes being stressed. In addition to the raised voltage and temperature, external signals are applied to the input pins that are wired in parallel for all chips. This technique requires full product functionality at burn-in conditions but improves the burn-in efficiency while costs are still moderate and parallel stressing of many parts is easily possible. Still no feedback from any chip during burn-in is available. Further improvement is achieved by the monitored dynamic burn-in. In this case some limited response monitoring is performed. Depending on the extent of monitoring, the parallel stressing is limited and the equipment becomes more expensive. However the escape rate is reduced;
- The most advanced type of burn-in is the *TDBI (Test During Burn-In)*. It applies full functional test patterns and full response monitoring for each individual chip. The advantage is that exact failure times and signatures can be determined as well as equipment or contact problems. Thus, burn-in escapes can be minimized and chips that were not exposed to burn-in voltage can be recycled. The individual monitoring of each chip limits the number of parts that can be stressed on one burn-in board and the required equipment makes this type of burn-in usually very expensive.

In this work we focus on dynamic burn-in, one of the most common in industries [8]. One of the main causes of burn-in escapes is a lack in burn-in patterns [6] that makes not all nodes adequately stressed.

Usually, those patterns consist of randomly generated patterns or of functional patterns generated to validate and verify the correctness of the circuit functionality. In dynamic burn-in, since the response of the circuit is not taken into account, the goal of the test patterns is to stress the device under high voltage and temperature conditions. In this context the ability of uniformly stressing all parts of the circuit is a key factor to reduce the burn-in escapes and to enhance the reliability of the final product.

The goal of this work is to automatically generate test patterns able to exercise all the possible transitions (form 0 to 1, and from 1 to 0) in a uniform way in all the nodes of the circuit under test. We focus on full-scan sequential circuits. The motivation of this work comes from the observation that with both random and functional patterns (e.g., stuck at patterns), some nets of the circuit usually are more stressed than others during the test session.

As explained in Section 3, with respect to a generic ATPG for transition faults we do not need to observe the circuit output during the test but we need to take into account the transitions performed by the combinational nodes while scanning the patterns in the scan chain.

3. Automatic Test Pattern Generation

Our Automatic Test Pattern Generation algorithm tries to generate a test pattern able to force a couple of transitions ($0 \rightarrow 1$ and $1 \rightarrow 0$) in each node of a Device Under Test (DUT). We will focus on full-scan circuits and, for the sake of simplicity, we will consider a single scan chain. Anyway, the approach is general and can be applied to circuits with multiple scan chains.

During test, full-scan circuits can operate in two different modes: (i) scan mode, and (ii) capture mode. In scan-mode a test pattern is serially loaded (scanned) into the flip-flops of the scan chains in order to initialize the memory elements to a given value and to compute the next state value. In capture mode a clock cycle is applied in order to store the pattern results again into the flip-flops whose values can be read by scanning out their content.

In a traditional ATPG, as for example pattern generation for stuck-at faults, the ATPG does not need to consider what happens to the combinational part of the circuit during the scan-in phase. In this situation, the logic gates connected to the scan chain perform transitions according to the value temporary scanned in the flip-flops. Those transitions are not considered since they do not affect the synchronous behavior of the circuit unless a clock cycle is applied (capture mode).

In case of dynamic burn-in the situation is completely different. All the transitions forced on the nodes of the circuit need to be considered since they give a contribution to the total stress of the circuit.

The proposed ATPG takes into account this problem by generating test patterns that consider the transitions forced into the circuit during the scan-in phase.

The proposed approach is described in detail in the following subsections.

3.1. ATPG Algorithm

The ATPG algorithm is the core part of the proposed pattern generation process. It is an iterative process that keeps track of the transitions performed by each node of the DUT and, at each iteration, tries to generate a new test pattern (TP) to force a new transition on one of the nodes of the circuit. In order to achieve a uniform distribution of transitions in the circuit, nodes are sorted according to the number of transitions performed at that time. The target node is selected in such a way that it is the one with the minimum number of performed transitions.

Each new TP is composed by values to be forced both on the primary inputs (PI) of the circuit and to be loaded in the flip-flops composing the scan chain.

While applying values to the PI is immediate, loading values in the scan chain is a serial process that requires several clock cycles. Besides transitions explicitly addressed by the TP, during the scan process additional transitions are generated. Those transitions need to be considered and to be kept under control in order to uniformly stress the circuit.

Since each TP usually includes a huge number of don't care values, an intelligent assignment of don't care bits in order to compress final patterns is a possible way to achieve this task. This operation is particularly complex when considering the values to be scanned in the scan chain.

To solve the problem, we introduce data structures and definition used in the algorithm:

- OGL is the *Ordered Gate List*, i.e. the list of gates in the circuit ordered by number of performed transitions on the primary output. The list is dynamically updated every time a new bit is scanned into the scan chain. Each element of this lists holds three information: (i) the present logical output value of the gate, (ii) the number of transitions already performed (*Performed_Transitions*) and, (iii) the number of transitions already calculated but not yet performed since the TP has not been already scanned into the circuit (*Transitions_To_Perform*);
- CTP is the *Compressed Test Pattern*. It represents the serial test pattern to be applied into the scan chain. At each moment of the ATPG process, the CTP represents the sequence of bits already scanned into the scan chain. Every time a new TP is calculated, the algorithm first checks whether that TP can fit into the CTP. In that case, the TP will be automatically applied to the DUT during the scan phase. If the CTP does not contain the TP, a new bit is added to the CTP with the "don't care" value. The circuit is therefore simulated and the transition status updated. At the same time, the TP that has been calculated but not scanned into the circuit is stored in the related OGL element. The process ends when each node has performed at least one transition;
- NI is the *Next Input (NI)* i.e. the next bit of the CTP to scan into the scan chain.

```
while exists G in OGL with #Transition == 0
{
  do {
    G = First element of OGL;
    AllTransitions=#Performed_Transitions(G) +
                   #Transitions_To_Perform(G);
    if (AllTransitions == 0) {
        P = Pattern Generation Function (G)
        if (P != NULL) {
            if (P already contained in CTM) {
               -- see Section 3.1.2
               break;
             }
        }
    }
    Update #Transitions_To_Perform(G);
  } while #Performed_Transitions (G) == 0
  NI = '-';
  Shift the scan chain; --(See 3.1.3)
  Simulate the circuit
  Update the transitions count
  Update the OGL
}
```

Figure 1: ATPG Algorithm

The test pattern produced by the algorithm in Figure 1 guarantee that each node in the DUT will perform at least one transition. Since our target is to force two transitions on each node (i.e. $0 \rightarrow 1$ and $1 \rightarrow 0$) the produced test pattern need to be completed by applying a reset of the scan chain elements. The reset will force the elements of the circuit to their original states and will guarantee that all the gates that performed an odd number of transitions will perform an additional transition.

The final test pattern is therefore composed by:

- Reset of the scan chain
- CTP
- Reset of the scan chain

3.1.1. Pattern Generation Function (PGF)

The Pattern Generation Function (PGF) is in charge of generating a test pattern able to force a transition on a node of the DUT. The ATPG algorithm calls it each time a new node needs to be stressed.

The generation of a pattern is based on well-known techniques for fault excitation in combinational ATPGs. The list of logic gates connected between the circuit inputs, the scan chain and the given gate is first

calculated. This process defines a tree of gates with the target gate as the root and the nodes connected to the scan chains and PI that can affects the value of the target gate as leaves.

The desired pattern can be generated by assigning proper values to the gates in the tree ('0', '1', '-') thanks to the knowledge of the current state of the target node.

3.1.2. Pattern Compression

The pattern compression process checks if a new generated TP can be mapped into the current CTP by an opportune assignment of don't care values. Without the compression step, N bits would be added to the CTP (where N is the scan chain length) for each new pattern with a considerable increase of the test pattern length.

The pattern compression works as follow: it first checks whether the TP is exactly contained in CTP. In this case the procedure ends (the CTP is already up to date). Otherwise it looks for a mapping of the TP into the CTP considering also don't cares values. If it finds a mapping scheme, the CTM is modified according to that scheme; otherwise new bits are added to the CTM.

Figure 2 shows an example. The TP is '-101-'. The algorithm tries to search exactly the same sequence inside the CTM. There is a conflict (circled values) when the value of a bit in TP is different from don't' care ('-') and it is different from the corresponding bit in CTP. In this case the don't care value is considered different from the '0' and '1' value.

In the example of Figure 2 the TP is not contained in CTP so the compression continues trying to map the TP into the CTP by properly assigning don't care bits (Figure 3). To check if the mapping is possible the mechanism is analogues to the previous one, the only difference is the definition of conflict situation.

In this case there is a conflict when the value of a bit in the TP is different from "don't care" and the corresponding value in the CTP is different from "don't care" and the two bits have different values.
In the example of Figure 3 during Step 3 it is possible to match the pattern by mapping the value of a "don't care" bit in the CTP. This is possible because in those positions there are no conflicts (circled values). In this case the compression is possible, the CTP is updated and the compression ends.

Step 1:	(-)	(1)	0	1	-						TP
	(1)	(0)	0	1	0	-	-	1	0	-	CTP
Step 2:		(-)	(1)	(0)	(1)	-					TP
	1	(0)	(0)	(1)	(0)	-	-	1	0	-	CTP
Step 3:			(-)	1	0	(1)	-				TP
	1	0	(0)	1	0	(-)	-	1	0	-	CTP
Step 4:				-	(1)	(0)	(1)	(-)			TP
	1	0	0	1	(0)	(-)	(-)	(1)	0	-	CTP
Step 5:					-	(1)	(0)	1	(-)		TP
	1	0	0	1	0	(-)	(-)	1	(0)	-	CTP
Step 6:						-	(1)	(0)	(1)	-	TP
	1	0	0	1	0	-	(-)	(1)	(0)	-	CTP

Figure 2: Pattern Matching

Step 1:	-	(1)	0	1	-						TP
	1	(0)	0	1	0	-	-	1	0	-	CTP
Step 2:		-	(1)	(0)	(1)	-					TP
	1	0	(0)	(1)	(0)	-	-	1	0	-	CTP
Step 3:			-	1	0	1	-				TP
	1	0	0	1	0	-	-	1	0	-	CTP

Figure 3: Pattern Compression

4. Experimental Results

In order to validate the proposed approach we performed a set of experiments. We used the ISCAS85 Combinational Benchmark Circuits [9] as target benchmarks. The ISCAS85 Circuits have been converted into full-scan sequential circuits by connecting each primary input and each primary output to a D-Flip Flop. All the flip-flops have been thus connected to form a single scan chain (see Figure 4). This configuration is the worst case from our point of view since the circuit has no primary inputs and the scan chain is the only way to apply patterns.

For each circuit we considered three different test patterns: (i) ATPG: patterns generated using the proposed ATPG algorithm, (ii) RANDOM: patterns randomly generated, (iii) FULL SCAN: stack-at faults patterns generated using Testgen by Synopsys™.

Concerning RANDOM patterns, a key point is the pattern length. We decided to use the same length of FULL SCAN patterns. FULL SCAN patterns are usually longer than the ones generated by our ATPG (see Table 1). We decided to use a long RANDOM pattern to guarantee transitions on each node of the DUT and, at the same time, by generating an elevated

number of transitions we can estimate how those transitions are distributed.

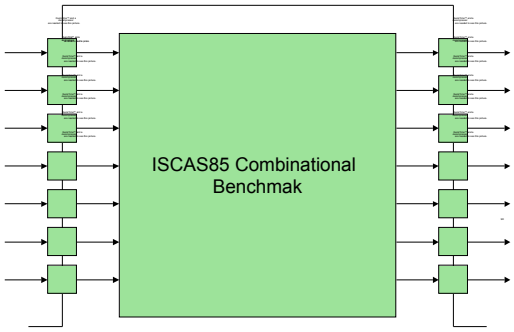

Figure 4: Benchmark Structure

To analyze the obtained results, for each selected circuit we performed a statistical analysis on the number of transitions executed by each node of different DUTs during the application of the three different patterns.

Table 1 shows the results of this analysis. For each DUT and for each type of test pattern we report: the overall number of nodes, the test pattern length, the number of total transitions executed during the pattern application, the average number of transitions performed by each gate and the variance from the average value.

Looking at the numbers of transitions and their average values for each type of pattern, it is possible to note that RANDOM and FULL SCAN patterns usually force an elevated number of transitions w.r.t the ATPG patterns. This is mainly due to the pattern length. FULL SCAN and RANDOM patterns are longer than the ATPG ones.

The length of the test pattern is not a problem during the burn-in test. In fact, in case of ATPG it is possible to increase the number of generated transitions by simply applying the same pattern repetitively during the burn-in period. The availability of a short pattern able to force at least one transition on each node repeated several times during the burn-in phase allows keeping under control the level of induced stress and also the level of power consumption by simply increasing or slowing the frequency at which the pattern is reapplied.

In addition, the uniformity of the induced transitions allows avoiding peaks of power consumptions in the circuit. The variance in Table 1 represents the uniformity of the generated test pattern.

By definition, the variance is a measure of the statistical dispersion of a variable. In our case it indicates how far from the average value the observed number of transitions per gate is. Our goal is to force the same number of transitions on each node of the DUT. The greater is the variance the greater is the difference in terms of number of transitions in the nodes of the DUT.

Looking at the results of Table 1 it is immediately clear that FULL SCAN and RANDOM patterns are much more dispersed then ATPG ones, i.e. they do not guarantee a uniform stress during the test.

Again the different length between the ATPG test pattern and the RANDOM and FULL SCAN test patterns is not a problem when comparing the variance. Actually to increase the length of ATPG test pattern it is enough to repeat the same sequence several times. In any case this operation does not modify the variance of the new test pattern since the distribution of the transition remains unchanged.

In addition, FULL SCAN and RANDOM patterns do not guarantee at least one transition on each node.

As an example in C880 using FULL SCAN patterns there are 26 nodes without transitions. The same happens in case of RANDOM patterns if we try to reduce the pattern length in order to increase the uniformity of the generated transitions. With RANDOM patterns of the same length of ATPG patterns, we obtained a comparable value in terms of variance but in C499 we obtained 19 nodes without transitions and in case C880 we obtained 36 nodes without transitions. The situation gets worse while increasing the circuit size.

C17					
PATTERN	*Nodes*	*Length*	*Trans.*	*Average*	*Variance*
ATPG	13	12	32	2,46	0,29
RANDOM	13	55	301	23,15	1,39
FULL SCAN	13	55	192	14,77	0,38
C499					
PATTERN	*Nodes*	*Length*	*Trans.*	*Average*	*Variance*
ATPG	275	133	6308	22,93	11,71
RANDOM	275	29643	3366970	12243,98	2299,17
FULL SCAN	275	29643	180068	654,79	257,30
C880					
PATTERN	*Nodes*	*Length*	*Trans.*	*Average*	*Variance*
ATPG	469	82	4468	9,52	3,62
RANDOM	469	20100	1635433	6300,23	525,27
FULL SCAN	469	20100	218175	465,19	308,72
C432					
PATTERN	*Nodes*	*Length*	*Tran.*	*Average*	*Variance*
ATPG	203	51	2278	11,22	2,95
RANDOM	203	12780	1035354	5100,00	458,00
FULL SCAN	202	12780	189233	936,80	111,19

Table 1: Transitions distributions

To conclude, Figure 5 graphically shows the distribution of transitions in case of the C499 benchmark. The x-axis reports the number of transitions whereas the y- axis reports the number of gates that performed that number of transitions. To compare the different approaches, the number of transitions has been normalized in order to have a distribution with average equal to zero. The graph clearly shows how RANDOM and FULL SCAN are more dispersed (i.e. far from the average value) than the ATPG approach.

5. Conclusions

This work proposed a full-scan circuit ATPG for dynamic burn-in. The goal of the proposed ATPG is the generation of test patterns able to force transitions into each node of full-scan circuits to guarantee to stress all the parts of the circuit during a dynamic burn-in test.

In addition, the proposed approach attempts to equalize the transitions forced into the circuit in order to avoid overstressing part of the device and possibly damaging it.

We successfully implemented the proposed algorithm and applied it on a set of four different ISCAS85 benchmarks. By comparing the obtained results and the ones obtained by using randomly generated patterns, or stack-at faults patterns generated using Testgen by Synopsys™ we have been able to stress the circuit in a more uniform manner with a shorter pattern. In addition our approach has been able to guarantee at least on transition on each node of the circuit in contrast with the other two approaches that do not guarantee this condition.

6. References

[1] T. Kim, W. Kuo, "*Modeling Manufacturing Yield and Reliability*", IEEE Transaction on Semiconductor Manufactoring, Vol. 12, No. 4, November 1999, pp. 485-492.

[2] E.R. Hnatek, "*Integrated Circuits Quality and Reliability",* Marcel Dekker, Inc., 1995.

[3] T. Barrette, V. Bhide, K. De, M. Stover, E. Sugasawara, "*Evaluation of Early Failure Screening Methods"*, International Workshop on IDDQ Testing 1997, pp. 14-17, 1996.

[4] T.Y.J. Chang, E.J. McCluskey, "*SHOrt Voltage Elevation (SHOVE) Test for Weak CMOS ICs*", IEEE VLSI Test Symposium 1997, VTS'97, pp. 446-451.

[5] Chin-Long Wey, Meng-Yao Liu "*Burn-In Stress Test of Analog CMOS ICs"* IEEE Asian Test Symposyum 2004, ATS'04, pp. 360-365

[6] D. L. Thompson, R. R. Wood, "Semiconductor Defect Reliability Modeling", IEEE IRPS Tutorials Notes 1996, p.p. 1a.1-1a.39.

[7] B.El-Kareh, W. R. Tonti, "Chip Reliability", IEEE IRPS Tutorial Notes 1995, pp. 3.1-3.56.

[8] R.P. Vollertsen, *"Burn-In"*, IEEE International Integrated Reliability Workshop Final Report, 1999, pp. 167 – 173.

[9] http://courses.ece.uiuc.edu/ece543/iscas85.html

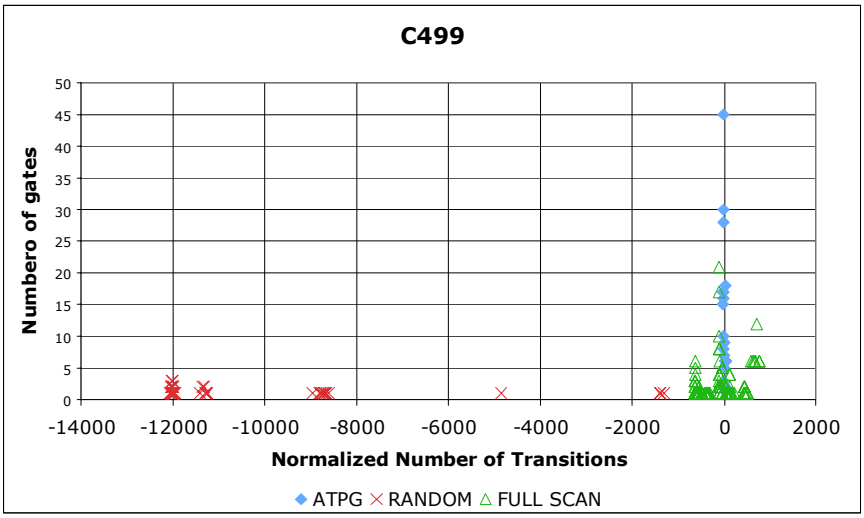

Figure 5: C449 Experimental Results