Testing SoC Interconnects for Signal Integrity Using Extended JTAG Architecture

Mohammad H. Tehranipour, Nisar Ahmed, Mehrdad Nourani

Center for Integrated Circuits & Systems
The University of Texas at Dallas
Richardson, TX 75083-0688
{mht021000,nxa018600,nourani}@utdallas.edu

ABSTRACT

As technology shrinks and working frequency reaches multi gigahertz range, designing and testing interconnects are no longer trivial issues. In this paper we propose an enhanced boundary scan architecture to test high-speed interconnects for signal integrity. This architecture includes: a) a modified driving cell that generates patterns according to multiple transitions fault model; and b) an observation cell that monitors signal integrity violations. To fully comply with conventional JTAG, two new instructions are used to control cells and scan activities in the integrity test mode.

Keywords: *Boundary Scan Test, Integrity Loss, Interconnect Testing, JTAG Standard, Signal Integrity, System-on-Chip.*

I. INTRODUCTION

A. Motivation

The number of cores in a system-on-chip (SoC) is rapidly growing which leads to significant increase in the number of interconnects. With fine miniaturization of the VLSI circuits, existence of long interconnects in SoCs and rapid increase in the working frequency (currently in gigahertz range), signal integrity has become a major concern for design and test engineers. Use of nanometer technology in SoCs magnifies cross-coupling effects among the interconnects. The coupling capacitance and mutual inductance affect the integrity of a signal by adding noise and delay. The noise effect can appear as overshoot and ringing. The former is known to shorten transistor lifetime and the latter to cause intermittent functional errors. Slowdown and performance degradation are often the result of excessive delay. Various parasitic factors such as parasitic capacitances, inductances and their cross-coupling effects on the interconnects, are difficult to control during fabrication. These parasitic factors play a significant role in the ultimate functionality and performance of high-speed SoCs.

Signal integrity is the ability of a signal to generate correct responses in a circuit. It generally includes all effects that cause the design to malfunction due to distortion of the signal waveform. According to this informal definition, a signal with good integrity has: (*i*) voltage values at required levels and (*ii*) level transitions at required times. If signal integrity losses (i.e., noise and delay) on an interconnect are within the defined safe margin, they are acceptable since they do no harm. Otherwise, they may cause intermittent logic-error, performance degradation, shorter life time and reliability concern. For example, an input signal to a flip-flop with good signal integrity arrives early enough to guarantee the setup and hold time requirements and it does not have spikes causing undesired logic transition (ringing).

The impact of process variation on circuit operation is an important issue in deep submicron (DSM). Process variation and manufacturing defects both may lead to unacceptable levels of noise and delay. The goal of design for DSM is to minimize noise and delay. However, it is impossible to check and fix all possible signal integrity problems during DSM design by only design rule checking (DRC), validation and analysis. Process variations and manufacturing defects may lead to unexpected changes in coupling capacitances and mutual inductances between interconnects. They in turn result in loss of signal integrity (e.g. glitches and excessive delay), which may eventually cause logic error and failure of the chip. The impact of spot defects and process variations on the magnitude of inductance induced noise are reported in [1]. The authors reported that the maximum crosstalk pulse considering process variation is almost twice the value for the nominal set of parameters. Since it is impossible to predict the occurrence of defects that cause noise and delay, signal integrity testing is essential to ensure error-free operation of the chip and must be addressed in manufacturing testing.

In recent years, various methodologies to test signal noise and skew on interconnects, due to different sources, are reported in literature Regardless of the method used to detect integrity loss, we also need a mechanism to coordinate activities in an integrity test session. We believe that one of the best choices is the boundary scan test methodology that includes the capability of accessing interconnects. Boundary scan test methodology was initially introduced to facilitate testing complex printed circuit boards. The IEEE 1149.1 Boundary Scan Test [5], also known as Joint Test Action Group (JTAG) Standard, has been widely accepted and practiced in the testing community. The standard provides excellent testing features with low complexity but it was not intended to address high-speed testing and signal integrity loss.

The standard, nevertheless, provides a mechanism to test core logic and the interconnects among them. Interconnects can be tested for stuck-at, open and short faults [6]. In this paper, the standard boundary-scan architecture is extended to test interconnects for noise and skew violation. While we focus on interconnects, any non-modeled fault (inside or outside cores) that manifests itself as integrity loss on interconnects will also be detected by our method.

B. Prior Work

•***Signal Integrity Modeling and Analysis:*** Maximum aggressor (MA) fault model [7] is one of the fault models proposed for crosstalk. Various approaches to analyze the crosstalk are described in [8] [9] [10]. Interconnect design for GHz+ integrated circuits is discussed in [11]. The author observed that chips failed, when a specific test pattern (not included in the MA model) is applied to the interconnects, due to overall effect of coupling capacitances and mutual inductances. Similarly, according to [12], the worst case switching pattern to handle inductive effects for multiple signal lines may not be included in the MA fault model. Several researchers have worked on test pattern generation for crosstalk noise/delay and signal integrity [13] [14] [15].

•***Test Methodologies:*** There is a long list of possible design and fabrication solutions to reduce signal integrity problems on the interconnect. None guarantees to resolve the issue perfectly. A double sampling data checking (DSDC) technique is used to capture noise-induced logic failures in on-chip buses [16]. At-speed testing of crosstalk in chip interconnects and testing interconnect crosstalk defects using an on-chip processor are reported in [3], respectively. A BIST (Built-In Self-Test) based architecture to test long interconnects for signal integrity [4], and the use of boundary scan and I_{DDT} for testing buses [17] are other proposed methods. Even short interconnects, especially those located near long interconnects, are also susceptible to integrity problems. Therefore, in the near future, methodologies are required for testing both short and long high speed interconnects [18].

•***Integrity Loss Sensor Cell:*** Due to increasing concerns about signal integrity loss in gigahertz chips and the fact that their occurrence must be captured on the chip, researchers presented various on-chip integrity loss sensors (ILS). A BIST structure using D flip-flops has been proposed to detect deviation of propagation delay in operational amplifiers [19]. In [17] a built-in sensor is integrated within the system. This sensor is an on-chip current mirror converting dissipated charges into associated test time. Reference [20] presented a more expensive but more accurate circuit to measure jitter and skew in the range of few picoseconds. The authors in [21] presented a sample and a hold circuit that probes the voltage directly within the interconnects. The work presented in [4] proposed two cells, called noise detector (ND) and skew detector (SD) cells, based on a modified cross-coupled PMOS differential sense amplifier. To detect delay violation, an integrity loss sensor (ILS) has been designed in [22] which is flexible and tunable for various delay thresholds and technologies. A double sampling technique is applied by on-line error detector circuit to test multiple-source noise-induced errors on the interconnects and buses [16].

•***Modified Boundary Scan and IEEE Standards:*** BIST-based test pattern generators for board level interconnect and delay testing are proposed in [23] and [24], respectively. A test methodology targeting defects on bus structures using I_{DDT} and boundary scan has been presented in [17]. P1500 proposes standardization of Core Test Wrapper and Core Test Language (CTL) [25]. This method is similar to boundary scan in terms of serial data transfer and therefore, the extended Serial Interface Layer (SIL) architecture can be used for various test applications at the system level in general and in integrity test, in particular.

IEEE 1149.4 mixed-signal test bus standard [26] was proposed to allow access to the analog pins of a mixed-signal device. In addition to the ability to test interconnects using digital patterns, 1149.4 includes the ability to measure actual passive components, such as resistors and capacitors. This standard cannot support high frequency phenomena such as crosstalk on the interconnects. Reference [27] proposes a method to simplify the development of a mixed signal test standard by adding the analog interconnect test to 1149.1. IEEE 1149.6 provides a solution for testing AC-coupled interconnects between integrated circuits on printed circuit boards and systems [28]. Our approach is similar to this standard draft in strategy of enhancing the JTAG standard and its instructions for testing high-frequency behaviors. However, there are fundamental differences. Contrary to our approach, the standard is not intended to consider coupling effects among the interconnect lines. Also, 1149.6 adds a DC blocking capacitor to each interconnect under test to disallow the DC signals. Thus, 1149.6 cannot test integrity loss due to low-speed, but very sharp-edge signals that are known to cause overshoots and noise. The sensors in our architecture, sitting in the observation boundary scan cells, can detect such scenarios. Finally, using differential drivers in the modified cells in 1149.6 makes the cells more expensive and less flexible in adopting other types of noise detectors/sensors. Various issues on the extended JTAG architecture to test SoC interconnects for signal integrity are reported in [22] [29] and [30] using the Maximum Aggressor (MA) and Multiple Transition (MT) fault model, respectively.

C. Contribution and Paper Organization

Our main contribution is an on-chip mechanism to extend the JTAG standard to include testing interconnects for signal integrity. The modified driving-end boundary scan cells (called PGBSC) receive a few seeds and generates Multiple Transition (MT) patterns at-speed to stimulate integrity violations. The MT pattern set is a superset of the MA set and is much more capable of testing the capacitive and inductive coupling among interconnects. The modified receiving-end boundary scan cells (called OBSC) record the occurrence of signals entering the vulnerable region over a period of operation. Using two new instructions in JTAG architecture, the integrity test information

is sent out for a final test analysis, reliability judgment and diagnosis.

The rest of this paper is organized as follows. Section II describes the MT fault model and its corresponding test patterns. The enhanced boundary scan cells are detailed in Section III. Section IV explains the test architecture to send test patterns and capture and read out the signal integrity information. The experimental results including implementation and simulation of our own integrity loss sensor are discussed in Section V. Finally, concluding remarks are in Section VI.

II. Multiple Transition (MT) Fault Model

The MA fault model [7] is a simplified model used by many researchers mainly for crosstalk analysis and testing. This model, shown in Figure 1, assumes the signal traveling on a victim line V may be affected by signals/transitions on other aggressor line(s) A in its neighborhood. The coupling can be represented by a generic coupling component Z. In general, the result could be noise (causing ringing and functional error) and delay (causing performance degradation). However, there is controversy as to what patterns trigger maximal integrity loss. Specifically, in the traditional MA model that takes only coupling C into account, all aggressors make the same simultaneous transition in the same direction, while the victim line is kept quiescent (for maximal ringing) or makes an opposite transition (for maximal delay). When mutual inductance comes into play, some researchers have shown that the MA model may not reflect the worst case, and presented other ways (pseudorandom, weighted pseudorandom or deterministic) to generate test patterns to create maximal integrity loss [13] [14] [15].

As reported in [11], a chip fails when the nearest aggressor lines change in one direction and the other aggressors in the opposite direction. This and many similar carefully chosen scenarios are not covered by the MA fault model. Exhaustive testing covers all situations, but it is very time consuming because of the huge number of test patterns. Additionally, exhaustive or pseudorandom patterns include some cases where aggressors are in quiescent mode and obviously do not maximally affect the victim line for noise and delay. Therefore, they need not be considered in model or pattern generators. Based on these observations reported by researchers, we define a new fault model and corresponding test set which covers *all* transitions on victim and *multiple* transitions on aggressors. The main idea behind our proposed fault model called Multiple Transition (MT) is having a single victim, a limited number of aggressors, a full transition on victim and multiple transition aggressors. In the MT model all possible transitions on the victim and aggressors are applied, while in the MA model only a subset of these transitions are generated. Another difference is single versus double direction change of aggressors in MA and MT, respectively. Briefly, the MA-pattern set is a subset of the MT-pattern set. Note carefully that the MT model is not an exhaustive model because it does not cover quiescent cases of aggressor lines for which integrity loss will not be obviously maximal.

As a motivating example, Figure 2 shows the simulation results of two MT-patterns (i) 0110110 $\rightarrow$ 1001001 and (ii) 0110101 $\rightarrow$ 1001010 and one MA-pattern i.e. 1110111 $\rightarrow$ 0001000 applied to a seven interconnect system while the middle wire is the victim and the others are aggressors. Extraction and simulation are done by OEA tool (BUSAN) [31] and TISPICE [32], respectively. We extracted the distributed RLC and coupling capacitances and mutual inductances using OEA tool for $0.18\mu m$ technology. As shown, MT-patterns create more delay as compared to MA-pattern, ranging from 35 to 70ps depending on the buffer size. Therefore, MA-patterns may not be able to generate maximum noise/delay on the victim line especially when inductance is included. In our opinion, this is the main reason why some researchers (e.g. [11]) reported scenarios in which test patterns not covered by the MA model failed a chip. Figure 4 shows all possible transitions on three line interconnects where the middle one is victim based on the MT fault model. Test patterns for signal integrity are vector-pairs. For example when the victim line is kept quiescent at '0' (column 1), four possible transitions on the aggressors are examined. The MA-patterns (a subset of MT-patterns) are shaded in Figure 4.

Four cases are examined for each victim line when victim

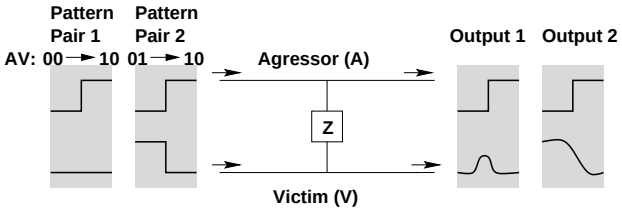


Fig. 1. Signal integrity fault model.

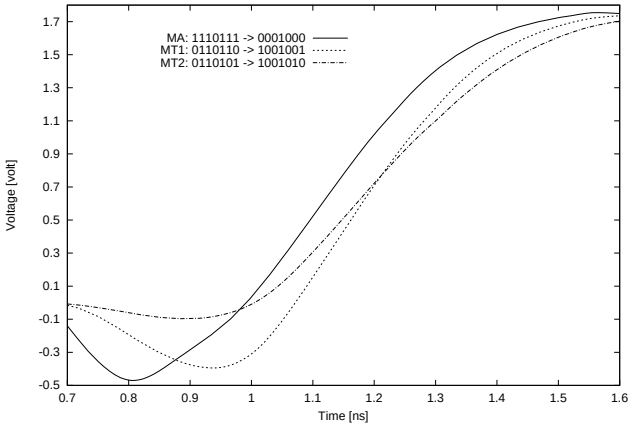


Fig. 2. Comparison between the MT and MA models.

	P_{g0}	P_{g1}	N_{g1}	N_{g0}	d_r	d_f
A	↑	↑	↓	↓	↑	↓
V	0	1	1	0	↓	↑
A	↑	↑	↓	↓	↑	↓
Vector 1: AVA	000	010	111	101	010	101
Vector 2: AVA	101	111	010	000	101	010

Fig. 3. The MA fault model and test patterns.

Quiescent at 0	Transition 0 → 1	Quiescent at 1	Transition 1 → 0
x0x x0x	x0x x1x	x1x x1x	x1x x0x
000 101	000 111	010 111	010 101
001 100	001 110	011 110	011 100
100 001	100 011	110 011	110 001
101 000	101 010	111 010	111 000

Fig. 4. All transitions on a three interconnect system that the MT and MA (shaded) models generate.

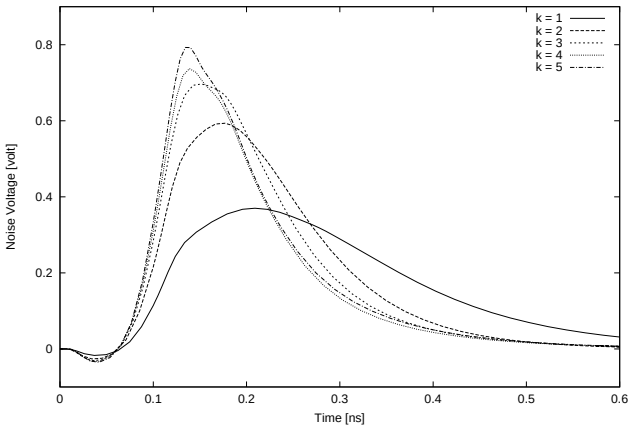


Fig. 5. Simulation results for different k.

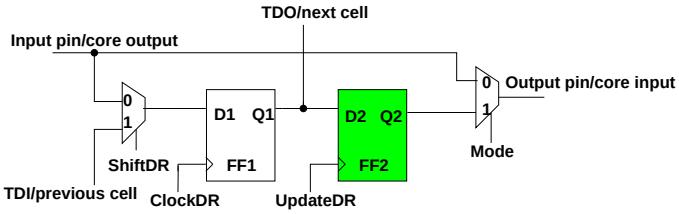


Fig. 6. A standard boundary scan cell.

line is quiescent at '0', '1', changes from '0' to '1' or '1' to '0'. As shown in Figure 4, the number of required test patterns to cover all possible transitions on the three interconnects is $4*2^{3-1}$=16 when the middle line is in victim mode. The total number of required test patterns is 3*16=48 when all three lines are examined in victim mode. In general, the number of test patterns for a group of m interconnects is $N_P = m \cdot 4 \cdot 2^{m-1} = m \cdot 2^{m+1}$. When m increases the number of test patterns increases exponentially. Simulations show that in an interconnect system the lines which are far away from the victim cannot affect the victim line much [11][15]. Therefore, the number of lines (aggressor) before and after the victim line can be limited. We define k as the *locality factor* that is empirically determined showing how far the effect of aggressor lines remain significant. Briefly, the total number of patterns to be generated will be $m \cdot 2^{m+1}$, where $m = 2k+1$. By choosing k (e.g. $k < n$ or $k << n$, where n is the total number of interconnects under test) user can do tradeoff between time and accuracy.

Figure 5 shows the simulation results, using OEA and TISPICE tools [31] [32], of a different number of aggressors in the victim neighborhood while the victim line is quiescent at 0 for $V_{dd} = 1.8V$. As shown, when the number of interconnects on either side of the victim increases the noise on the victim increases. The glitch height and period mainly depend on the driver strength. As the driver strength increases, the glitch height increases and the period decreases. The peak noise difference corresponding to two locality factors of k=3 and k=4 is $V_{peak}(k=4) - V_{peak}(k=3)$=0.048v which, for many applications, can be assumed negligible. If that is the case, k=3 can be used as the locality factor in the simulation and pattern generation. The percentage difference between glitches of different distributions with respect to k decreases with an increase in driver strength [7]. We acknowledge that finding such a locality factor is technology-dependent (e.g. parasitic RLC values) and application-dependent (e.g. depending on driven core or shielding techniques). However, once provided based on the application and accurate simulation, the total number of patterns and time to test integrity faults will be significantly reduced. Our observation is that, for current technologies and simulation based on the reduced order models, a small k (often less than 4) can produce accurate behavior of the interconnects. In any case by choosing k the user can always tradeoff between longer simulation time and accuracy.

III. Enhanced Boundary Scan Cells

Boundary scan is a widely used test technique that requires boundary scan cells to be placed between each input or output pin and the internal core logic. The standard provides an efficient mechanism for functional testing of core logic and interconnects. Figure 6 shows a conventional standard boundary scan cell (*BSC*) with shift and update stages. $Mode = 1$ puts the cell in test mode. The data is transferred into the shift register (in *Shift-DR* state) during scan operation. Test patterns scanned into the boundary scan cells through the scan-in port (*TDI*) are applied in parallel in *Update-DR* state (*UpdateDR signal*). Circuit response is captured in parallel by the boundary scan cells connected between internal logic and output pins, and is scanned out through the scan-out port (*TDO*) for observation [5].

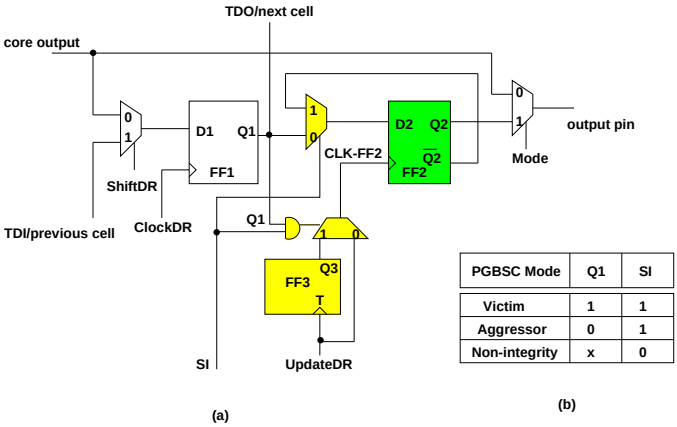
Using the JTAG standard (*IEEE 1149.1*), interconnects can be tested for stuck-at, open and short faults. This is possible by *EXTEST* instruction, by which the TAP controller isolates the core logic from the interconnects using the BSCs. However, *EXTEST* was not intended to test interconnects for signal integrity. We propose new cells and instructions for signal integrity testing. For this purpose, some minor modifications are applied to the standard architecture to target interconnects for signal integrity.

A. Pattern Generation BSC (PGBSC)

Analysis of the MT fault model test vector-pair shows that in some transitions the victim line should remain quiescent while the aggressor lines change. In some other transitions, both the victim and aggressor lines change. Additionally, in all cases

Seed	Quiescent at 0 x0x x0x	Transition 0 → 1 x0x x1x	Quiescent at 1 x1x x1x	Transition 1 → 0 x1x x0x
000	000 101	101 010	010 111	111 000
001	001 100	100 011	011 110	110 001
100	100 001	001 110	110 011	011 100
101	101 000	000 111	111 010	010 101

Fig. 7. Resorted test pattern for a three interconnect system.



PGBSC Mode	Q1	SI
Victim	1	1
Aggressor	0	1
Non-integrity	x	0

Fig. 8. PGBSC structure.

the aggressor lines change from one value to another ('0' to '1' or '1' to '0') at every clock, while the victim line value changes every two clocks. This important observation helps us design a circuit to efficiently generate MT test patterns. Figure 7 shows test vectors of Figure 4 resorted to make the point clear. Each row clearly shows that the victim changes every two clocks and the aggressors change every clock.

Each row in Figure 7 needs one seed. For example, for seed='000', these test patterns (shown in the first row) are generated: ('000','101'), ('101','010'), ('010','111') and ('111','000'). To cover all possible transitions as shown in Figure 7, four seeds are required. In the above three interconnect system, these seeds are '000', '001', '100' and '101'. The total number of required seeds to cover all the lines in victim mode in a three interconnect system is 3*4=12. For a group of m interconnects, where $m = 2k+1$, the total number of seeds are $N_S = m \cdot 2^{m-1} = (2k+1) \cdot 2^{2k}$, in which 2^{m-1} shows all the possible combinations of $m-1$ aggressor lines. Note carefully that as we discussed in Section II, since the locality factor k is very small, the total number of seeds remains very limited. For example, two seeds are sufficient to stimulate maximum delay in the interconnect according to the results in [12].

The MT test patterns cover the MA test patterns. In Figure 7, the shaded patterns show MA patterns which are generated based on only two seeds. It shows that after applying the first seed, '000', the generated test patterns cover the P_{g_0}, d_f, and P_{g_1} faults. The generated test patterns after applying the second seed, '101', cover the d_r, N_{g_0} and N_{g_1} faults (see Figure 3). Therefore, by such reordering, two seeds are sufficient for covering all 12 test patterns in the MA fault model.

A pair of test vectors are required to test the interconnects for signal integrity. These patterns can be applied to the interconnects using conventional boundary scan architecture. To apply each pair, the first and second patterns are scanned into the conventional BSCs and stored in FF_2 and FF_1, respectively (see Figure 6). Then, using *UpdateDR*, they are applied to the interconnects. Scanning and applying patterns in this way are very straightforward but need a large number of clocks which increase the overall test time. We propose a hardware-based method for test pattern generation for the MT fault model. Test pattern generation is performed at the input side of interconnects, that is the output side of a core which drives the interconnects. The new BSC that generates test patterns is called pattern generation BSC (PGBSC).

Boundary scan cell can be utilized to support the MT model. FF_2 in conventional boundary scan cell (shaded flip-flop in Figure 6) can be used to generate test patterns based on a given seed. FF_1 is used to initialize FF_2. A T flip-flop divides the clock in half and *UpdateDR* plays the same role as the clock in driving FF_2 (see Figure 8). First, the seed comes from TDI and is stored in FF_2 through FF_1. Victim/aggressor select signals are then scanned into FF_1 through TDI to select the victim and aggressor lines. Note carefully that only the seeds need to be scanned in instead of exact test patterns. This significantly reduces the number of required clock cycles for applying test patterns to the interconnects in a system-chip.

In addition to its normal (non-integrity) mode, PGBSC should work as victim or aggressor in signal integrity test mode. The PGBSC architecture is shown in Figure 8(a). Note that additional components in PGBSC are located in the test path and therefore no additional delay is imposed on the normal mode. Only one extra control signal (SI) is needed in this architecture. The SI signal is generated by a new instruction, to be explained in Section IV. PGBSC generates required test patterns to cover the MT fault model. Figure 8(b) shows the operation modes of the PGBSC. Depending on the select line of the MUX attached to FF_3, this architecture has three modes:

1) **Victim Mode:** Q_3 is selected. *UpdateDR* is divided by two and applied to FF_2. By every two *UpdateDRs*, the complemented data is generated in $\overline{Q_2}$ and it is transferred to the output pin.
2) **Aggressor Mode:** *UpdateDR* is selected, but PGBSC is in signal integrity mode. *UpdateDR* is applied to FF_2. By each *UpdateDR*, the complemented data is generated in $\overline{Q_2}$ and it is transferred to the output pin through feedback and Q_2.
3) **Non-Integrity Mode:** *UpdateDR* is selected. It is the normal test mode of the PGBSC (e.g. conventional boundary scan test) and *UpdateDR* is applied to FF_2.

Figure 9 shows the operation of a PGBSC. If PGBSC is in victim mode, *UpdateDR* is divided by two and generates *CLK-FF2*. If the initial value in Q_2 is '0', then $\overline{Q_2}$ is '1' and is applied to D_2 through the feedback. Every two activations of *UpdateDR*, the content of FF_2 is complemented. On the

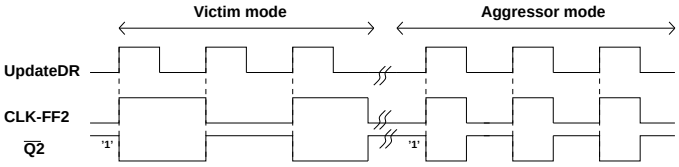


Fig. 9. Operation of the PGBSC.

TABLE I

VICTIM-SELECT DATA IN A *n-bit* INTERCONNECT SYSTEM WHEN $k = 2$.

Victim Location	Victim-Select Data
VAAVAA...VAA	100100...100
AVAAVAA...VA	0100100...10
AAVAAVAA...V	00100100...1

other hand, if PGBSC is in aggressor mode, *CLK-*FF_2 has the same frequency as *UpdateDR* and by each *UpdateDR* the content of FF_2 is complemented. As shown in Figure 8, $\overline{Q_2}$ is complemented by each *CLK-*FF_2 while Q_2 is applied to the interconnect.

Each interconnect can act as a victim or an aggressor. Therefore, in each test session the victim interconnect should be specified. After victim is tested, it will become an aggressor for other new victims. Briefly, for complete interconnect testing, the victim line rotates. One of the major advantages of limiting the number of aggressor lines is that parallel testing of interconnects becomes possible. In other words, in each step of the test we deal with at most k lines as aggressors before and k lines after the victim line. We use encoded data, called *victim-select data*, to specify the victim.

Table I shows the victim-select data for a n-bit interconnect system, to be scanned and stored in FF_1 when the locality factor is k=2. After specifying the victim, test vectors are generated by PGBSC and applied to the interconnects. Then, the new victim line is specified and the process will be repeated for the new victim. As shown in Table I, when we scan '100100...100' into n PGBSCs, the first line is the victim and the next two lines are aggressors for the first interconnect as well as for the fourth one. Therefore, $\lceil n/(k+1) \rceil = \lceil n/3 \rceil$ victims are tested simultaneously. As shown, with one clock the victims locations rotates ('0100100...10'). For $k = 2$ two rotations (clocks) are enough to set victim/aggressor nets and cover the entire interconnect.

The generic behavior of test pattern generation and applying procedure is shown in Figure 10. This behavior will be executed by a combination of the automatic test equipment (ATE) and the TAP controller. The first seed is applied to new BSCs as an initial value into FF_2. Victim and aggressors are selected with the victim-select data scanned into FF_1, and then the cells are set in SI mode to start generating test patterns. After generating test vectors and applying them to interconnects, a new victim is selected and the process will be repeated with the same initial value. Note carefully that at the end of testing one victim again (line 8) the same seed would be in FF_2 (See also Figure 7 for an example). Therefore, we do not need to scan in the seed for the next victim. This significantly reduces the overall test time. The same process

```
01: for (i= 1 to N_S)
02: {
03:       Scan seed i into FF_2
04:       Activate signal integrity test mode (SI=1)
05:       Scan the first victim-select data
06:       For (j=1 to k)   //Total # of shifts for victim-select data
07:       {
08:            Apply 4 UpdateDRs.          // Pattern generation
09:            Shift one '0' into FF_1     // Selecting new victim
10:       }
11: }
```

Fig. 10. Test pattern generation procedure using PGBSC.

will be repeated for all the seeds.

• ***Comment on External Test Patterns***

PGBSC is implemented based on a slight modification of the boundary scan cell. The MT test patterns are generated and applied by PGBSC. can be also efficiently used for external (deterministic or weighted pseudorandom) patterns. These patterns are applied by an external tester through TDI input pin. More specifically, PGBSC cells provide extra control and flexibility to scan in patterns and apply them to interconnects, and they even mix the MT model with other fault models. The flexibility of a PGBSC cell can be further explored for various tradeoffs. For example, a simple yet efficient compression technique for the enhanced boundary scan architecture using the PGBSC cells to minimize delivery time is discussed in [22].

B. Observation BSC (OBSC)

We propose a new BSC at the receiving side of interconnects which can employ any integrity loss sensor (ILS) such as the one presented in [22]. Figure 11(a) shows the new BSC, named observation BSC (OBSC). As shown, the ILS cell adds to the receiving cells, and captures signals with noise and delay at the end of an interconnect. When it receives a signal with an integrity problem (noise or skew violation) it produces a pulse at its output and the FF is set to '1'. The ILS is activated by the signal cell enable (*CE = '1'*). If *CE = '0'*, the ILS is disabled but captured data in its flip-flop remain unchanged. The OBSC operates in two modes as summarized in Figure 11(b).

1) **ILS Mode:** ILS flip-flop is selected. In this case, every time the *shiftDR* signal becomes active the captured ILS data is scanned through the scan chain for final evaluation.
2) **Non-Integrity Mode:** In this mode, ILS is isolated and each OBSC acts as a conventional (non-integrity) BSC.

In SI test mode, as Figure 11 shows, ILS FF can be read and scanned out for final evaluation. Before starting the scan out process, we need to send the content of ILS FF to FF_1. In this case, *sel* should be zero. Therefore, the SI and *ShiftDR* signals should be '1' and '0', respectively. When the scanning out process starts, D_1 is transferred to Q_1 to be used as a TDI for the next cell. After sending the value of ILS FF to Q_1, a scan chain must be formed. In this case, during the *Shift-DR* state, TDI input must be connected to FF_1.

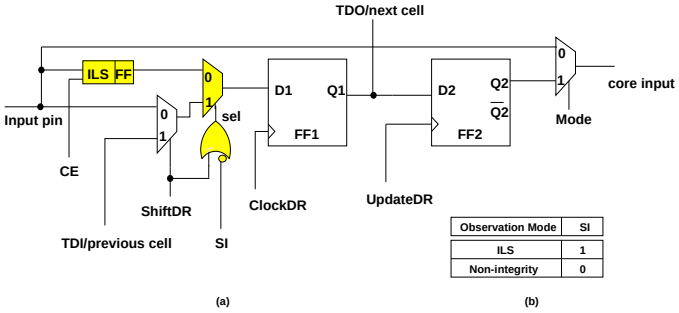


Fig. 11. OBSC structure.

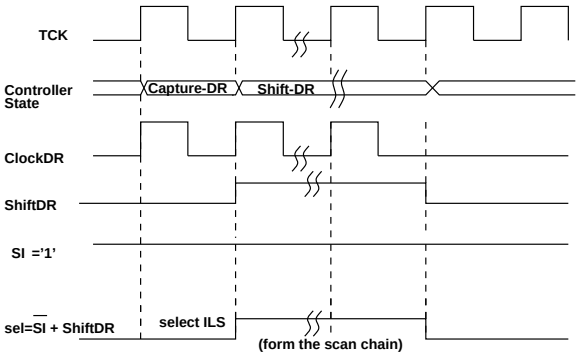


Fig. 12. Operation of the OBSC.

Therefore, the ILS path should be isolated by *sel*='1' (*SI*='1' and *ShiftDR*='1'). As shown in Figure 11, $\overline{SI}$ and *ShiftDR* are ORed together for selecting the ILS path for transferring ILS *FF* to D_1 and scanning out data in the chain. Figure 12 shows the dependency of *sel* on *SI* and *ShiftDR*. As shown, in *Capture-DR* state, ILS *FF* is selected and then in the *Shift-DR* state a scan chain is formed and data is scanned out depending on how many wires are under test. Table II shows the truth table of signal *sel*. Additional control signals (i.e. *SI* and *CE*) are generated by a new instruction, to be explained in Section IV. There are three methods of observation:

1) **Method 1:** To capture and scan out ILS data only once after applying the entire test pattern set and covering all victims.
2) **Method 2:** To capture and scan out ILS data N_S times, after the application of a group of patterns (e.g. the patterns generated based on a specified seed) for each victim.
3) **Method 3:** To capture and scan out ILS data right after applying each test pattern.

The first method has the least test time and a disadvantage for not being able to determine which transitions have caused the fault in an interconnect. The second method provides

TABLE II
TRUTH TABLE OF SIGNAL *sel*

SI	*ShiftDR*	*sel*
1	0	0
1	1	1
0	x	1

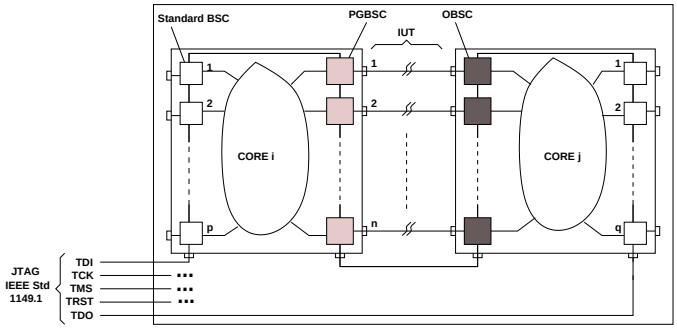


Fig. 13. Test Architecture

more information, to determine which set of transitions or faults caused the violation in the interconnects, at the expense of more test time. Finally, the third method is the most informative one, but it is extremely time consuming. In Section V we will report the experimentation and compare these methods. Although power issue is beyond the scope of this paper, we would like to comment that extra switching activities due to the MT model will be confined in boundary scan cells. In other words, as Figure 11 shows, cores sitting on the two sides of interconnects are isolated in signal integrity test mode and do not suffer from excessive power.

IV. TEST ARCHITECTURE

Figure 13 shows the overall test architecture with *n* interconnects between Core *i* and Core *j* in a two-core SoC. The five standard JTAG interfaces (*TDI*, *TCK*, *TMS*, *TRST* and *TDO*) are still used without any modification. The Test Access Port (TAP) controller and instruction register provide various control signals to boundary scan cells (Figure 6). The mandatory instruction PRELOAD is initially used to scan data into boundary scan cells. Two new instructions are defined for signal integrity test, one to activate PGBSCs to generate test patterns and the other to read out the test results. As shown in Figure 13, the cells at the output pins of *Core i* are changed to PGBSCs and the cells at the input pins of *Core j* are changed to the OBSCs. The remaining cells are standard BSCs which are present in the scan chain during signal integrity test mode. In the case of bidirectional interconnects, boundary scan cells used at both ends are a combination of PGBSC and OBSC to test the interconnects in both directions.

The ILS part of OBSCs does not need any special control and automatically captures the occurrence of integrity loss. After all patterns are generated and applied, signal integrity information stored in the ILS FF is scanned out to determine which interconnect has a problem. In Subsection III-B, we have presented various methods to readout these information. For example, Method 1 can be used to minimize the overall test application time since the information in the cells is scanned out once per interconnect as opposed to once per pattern.

A. Clock Analysis

In this section, we analyze the required clocks to apply the test patterns and read out the test results for different methods.

TABLE III
PATTERN GENERATION AND APPLICATION CLOCK ANALYSIS

Test Architecture	N_{clk}
Conventional BSC	$m \cdot N_P(n+4)+p$
Enhanced BSC	$N_S(2n+8k)+p$

TABLE IV
CLOCK ANALYSIS OF THE READOUT METHODS.

Readout Methods	N_{clk}
Method 1	n+2q
Method 2	$N_S \cdot n$+2q
Method 3	N_P(n-1)/2+2q

Assume that n is the number of interconnects under test, p is the number of boundary scan cells in the scan chain before the first PGBSC and q is the number of boundary scan cells in the scan chain after the last OBSC (see Figure 13).

In conventional BSC, test patterns are scanned in and applied to the interconnects one-by-one. In a n-wire interconnect network, the number of required clocks to scan and apply the MT test patterns by conventional boundary scan architecture through TDI is $N_{clk\_BS} = m \cdot N_P(n+4)+p$, where N_p is the total number of MT-patterns and $m = 2k+1$. When PGBSC is used, only seeds are scanned into the cells and the other test patterns are generated internally. Having n interconnects, the required clocks in the enhanced boundary scan method is $N_{clk\_EBS} = N_S(2n+8k)+p$, where k is the locality factor defined in Section II. This formula essentially counts clocks when the architecture executes the test generation algorithm shown in Figure 10. Tables III and IV summarize the exact number of required clocks for test generation/application and readout, respectively. Comparing conventional and enhanced boundary scan, test application time reduction (TR) will be:

$$TR\% = \frac{N_{clk\_BS} - N_{clk\_EBS}}{N_{clk\_BS}} \cdot 100\% \qquad (1)$$

B. New Test Instructions

IEEE 1149.1 allows the user to define some optional instructions. We propose to add two new instructions G-SITEST and O-SITEST to the mandatory instruction set. These new instructions are only used in signal integrity mode for testing interconnects.

- ***G-SITEST Instruction***

This instruction is used for test pattern generation using the enhanced boundary scan architecture. Seeds typically would be loaded onto the boundary scan cells using the PRELOAD instruction before loading the G-SITEST instruction. Thus, when the G-SITEST instruction takes place in *Update-IR* controller state, the related signals will be activated. G-SITEST targets PGBSCs and enables *SI*=1 during execution of instruction. It also enables the ILS cells (i.e. *CE*=1) to capture signal integrity information. The victim-select data is then shifted into FF_1 of PGBSCs during the *Shift-DR* state and patterns for the MT fault model are generated every *Update-DR* state as explained in Section III-A. Three *UpdateDRs* are required to generate three test patterns per victim line for each initial value. Each boundary scan cell at the output of a core whose interconnect is under test will behave according to G-SITEST instruction. The flow of data through PGBSC is shown in Figure 14. *Core i* executes the PRELOAD instruction and the cores before *Core i* execute the BYPASS instruction to scan in initial data (seeds) from the system input. The seeds are scanned in (broken lines in Figure 14) while other necessary changes to generate patterns are done internally using additional components (shaded components in Figure 14) in the PGBSC cell.

- ***O-SITEST Instruction***

This instruction is loaded after execution of the G-SITEST instruction and when the loss information has been stored in ILS FF. Assuming all test patterns have been applied to interconnects, O-SITEST instruction is used to capture and scan out the content of ILS FF. After the instruction is decoded in the *Update-IR* state, control signals *SI*=1 and *CE*=0 (to deactivate ILS) are generated. With *CE*=0, ILS will not receive input data anymore as the O-SITEST instruction is executed. The flow of data through the OBSC cell is shown in Figure 15. All cores after *Core j* execute the BYPASS instruction to scan out data to the system output.

Having two instructions in integrity test mode is necessary in enhanced boundary scan architecture. G-SITEST enables PGBSC to generate and apply test patterns and simultaneously, OBSC captures signals at the receiving-end of interconnects. After test application is performed, reading out the integrity information is begun by using the O-SITEST instruction. These two different operations cannot be done by only one instruction because FF_1 is reused for different purposes. Specifically, the content of ILS FF which is captured into FF_1 and then shifted to the next cells FF_1 will be overwritten by the next data captured by ILS FF cells during *Capture-DR* state in the next test pattern application cycle.

Figure 16 summarizes the test process and instructions used in signal integrity mode. First, the seed is scanned into boundary scan cells using the PRELOAD instruction. By loading the *G-SITEST* instruction into the instruction register, the new BSCs which target signal integrity test for interconnects are set in signal integrity mode. *Update-IR* makes the G-SITEST instruction operational and sets *SI=1* and *CE=1*. Then, test patterns are generated by PGBSCs and applied to interconnects and, simultaneously, ILS captures the signals at the end of interconnects and detects violations, if any. After the test application process, integrity loss data is stored in ILS FF and must be read out. This is done using the O-SITEST instruction. The instruction first deactivates the ILS cell because during the scan out operation some new data will be scanned in and it may be applied to interconnects in the *Update-DR* controller state before the scan-out phase is complete. This may cause ILS to lose integrity loss data stored in the previous cycle. Finally, the scanning out process is performed as explained in Section IV-B.

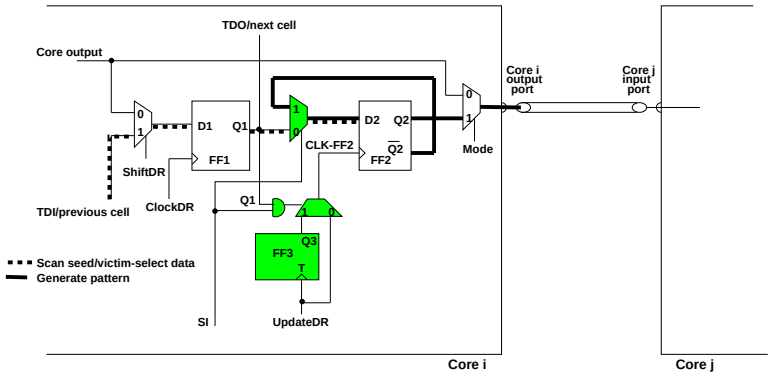


Fig. 14. Test data flow while the G-SITEST instruction is executed.

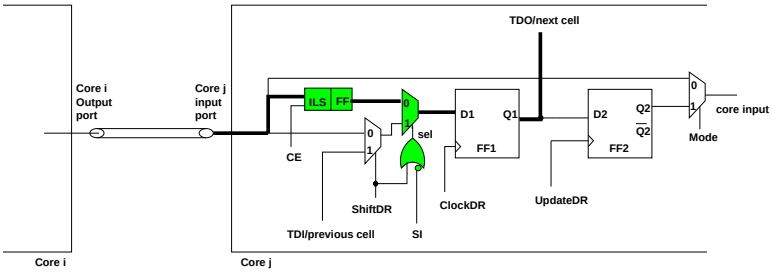


Fig. 15. Test data flow while the O-SITEST instruction is selected.

```
01: RESET
02: FOR (j=1 to N_S)
03: {
04:       PRELOAD seed j    // Initialize the PGBSCs
05:       G-SITEST    //Pattern generation and application
06:       FOR each victim line
07:             Apply 3 UpdateDRs
08: }
09: O-SITEST    //Reading out the integrity loss data
```

Fig. 16. Signal integrity test process.

V. Implementation and Simulation Results

A. ILS Cell Used in Our Work

While any ILS sensor, such as those presented in [4] [17] [19], can be used for the purpose of integrity loss detection, for the purpose of simplicity, cost and experimentation, we have developed our own ILS cell. In what follows, we briefly explain the functionality and characteristics of this cell.

Our ILS is a delay violation sensor shown in Figure 17. The cell consists of two parts, the sensor and the detector. The acceptable delay region (*ADR*) is defined as the time interval from the triggering clock edge during which all output transitions must occur. The test clock is used to create a window which determines the acceptable skew region. Signal input a is in the acceptable delay period if its transition occurs during the period when b is at logic '0'. Any transition that occurs during the period when b is at logic '1' is passed through the transmission gates to the detector, XNOR gate. The XNOR gate is implemented using dynamic precharged logic. Output c becomes 1 when a signal transition occurs during $b = 1$ and remains unchanged till $b = 0$, the next precharge cycle. Output c is used to trigger a flip-flop.

Figure 18 shows timing behavior of different input signals to the cell. Transmission gates are closed when signal $b = 0$ and their outputs hold to the previous values. Output of the ILS cell is determined by the timing of signals a and X. Three different cases for transition on signals a and X are illustrated as follows,

1) **Case 1:** Transition on signals a and X occurs during $b = 0$, i.e., inside the period t_{Inv1}. The transmission gate outputs change simultaneously and the XNOR gate output c is at logic '0'.
2) **Case 2:** Signal a makes a transition when $b = 0$ and X changes when $b = 1$. The transmission gate outputs have a delay difference of $(t_a + t_{Inv2}) - (t_{nand} + t_{Inv1})$. Signal c changes to '1' when this delay difference is greater than or equal to the XNOR gate delay. Equation 2 determines the minimum delay of signal a from the clock edge detected by the cell, i.e, the ADR period.
3) **Case 3:** The transition on signals a and X occurs during $b = 1$, i.e., outside the period t_{Inv1}. The transmission gate outputs have a delay difference equal to t_{Inv2} and the XNOR gate output c changes to logic '1'.

$$ADR = t_{a,min} = t_{XNOR} + t_{Inv1} + t_{nand} - t_{Inv2} \quad (2)$$

Equation 2 has two parameters t_{Inv1} and t_{Inv2} which can be adjusted to tune the cell to detect required delay. These two parameters are independent of each other. The minimum detectable delay ($t_{a,min}$) can be decreased by increasing t_{Inv2} or decreasing t_{Inv1}. t_{Inv2} can be decreased until the duration is enough to precharge the dynamic logic. Table V shows the minimum delay detectable by changing the number of cascaded inverters with delay of t_{NOT}, each to form Inverter 1 and 2. We acknowledge that depending on the application and

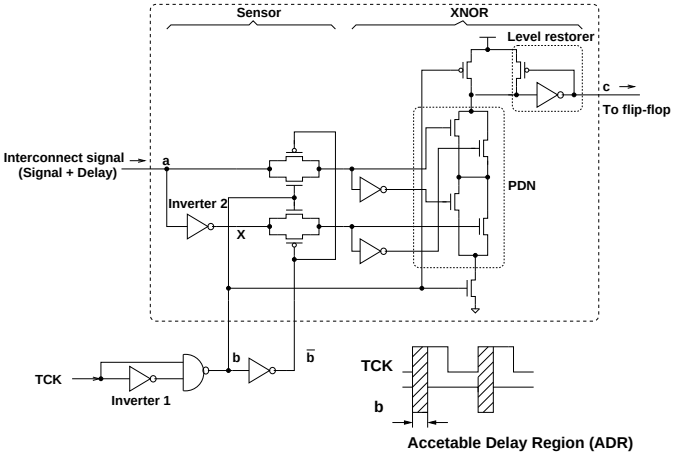


Fig. 17. Integrity loss sensor (ILS) cell.

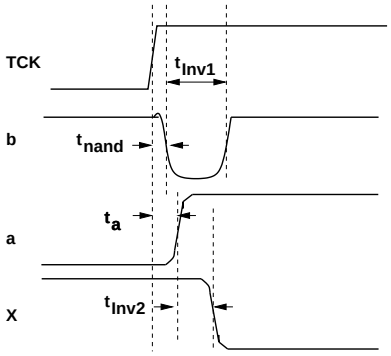


Fig. 18. ILS signals and their timing behavior.

threshold for delay and noise, we may need to re-tune ILS cells for different designs. Fortunately, most of sensors [4][17][20] are easily tunable. When needed, the delay of Inv2 can be tuned to control *ADR* for individual interconnects.

Figure 19 shows the SPICE [32] simulation of the ILS cell, implemented using 0.18 μm technology, for two transitions of signal at input a. The first transition of the signal occurs at 0.2 ns when $b = 0$ and the output remains zero. The second transition occurs at 3.5 ns when $b = 1$ and exceeds the acceptable delay period and output c goes to 1 until b goes to zero. This delay sensor can detect any transition faults (e.g. due to crosstalk) that result in excessive delay. The pulse produced on c can be fed to a flipflop to store delay occurrence for further readout/analysis.

B. Simulation Results

The enhanced boundary scan cell and architecture is implemented using Synopsys synthesizer [33]. The total area overhead for conventional BSC cell and the enhanced BSC

TABLE V

MINIMUM DETECTABLE DELAY OF ILS CELL.

Inverter1 (t_{Inv1}) [$\times 10t_{NOT}$]	*Inverter2* (t_{Inv2}) [$\times t_{NOT}$]	*Minimum Detectable Delay (ADR)* [ps]
5	1	450
3	3	250
1	5	100

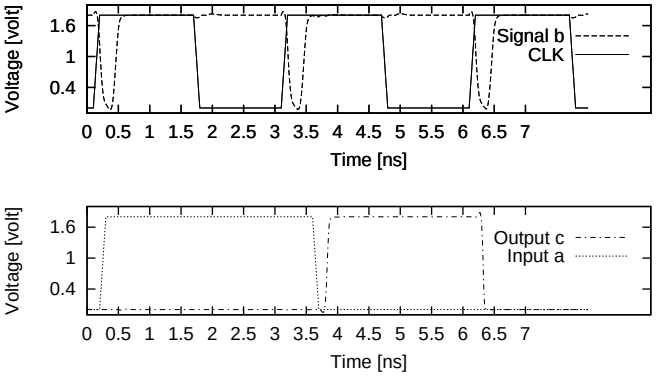


Fig. 19. Spice simulation of the ILS cell.

TABLE VI

COST ANALYSIS FOR BOUNDARY SCAN CELLS.

Test Architecture	*Cost[NAND]*		
	Sending	*Observing*	*Bidirectional*
Conventional Cells	26	26	78
Enhanced Cells	36	38	100
Area Increase %	38.5	46.2	28.2

cell (including ILS) is shown in Table VI. Enhanced cells are 28-46% more expensive compared to the conventional one. This is negligible compared to the overall cost of boundary scan architecture (cells, controller, etc.). Moreover, to lower the cost the user still has the option of inserting/changing cells on only interconnects of interest (e.g. those that are susceptible to noise or those that feed glitch/delay sensitive circuits).

Table VII shows the comparison between using enhanced boundary scan to generate and apply MT-patterns and conventional boundary scan to scan in and apply the same set of patterns. The application time reduction ratio is between 86 to 92% for various scenarios. Table VIII compares three methods described in Subsection III-B for different number of interconnects under test (n) and locality factor (k) assuming q=0. As expected, the number of clocks required for Methods 1 is significantly lower than Method 3. However, Method 3 provides much more information about the type and location of integrity faults. In Method 2, we have performed one scan-out operation per victim line. Essentially, Method 2 can be used to tradeoff test time versus accuracy.

TABLE VII

MT-PATTERN APPLICATION TIME.

Methods	*MT-Pattern Application Time [Cycle] (p=0)*					
	n=8		*n=16*		*n=32*	
	k=2	*k=3*	*k=2*	*k=3*	*k=2*	*k=3*
N_{clk_EBS}	2560	17920	3840	25088	6400	39424
N_{clk_BS}	19200	150528	32000	250880	57600	451584
TR%	86.1%	88.5%	88.3%	90.3%	88.9%	91.8%

TABLE VIII

OBSERVATION TEST TIME FOR THREE METHODS.

Methods	*Observation Test Time [Cycle] (q=0)*					
	n=8		*n=16*		*n=32*	
	k=2	*k=3*	*k=2*	*k=3*	*k=2*	*k=3*
Method 1	8	8	16	16	32	32
Method 2	640	3584	1280	7168	2560	14336
Method 3	2560	14336	5120	28672	10240	57344

TABLE IX
MT AND MA COMPARISON.

k	*MT*		*MA*	
	$V_{noise}[V]$	$Delay[ps]$	$V_{noise}[V]$	$Delay[ps]$
k=2	0.351	470.5	0.348	468.5
k=3	0.408	472.3	0.404	450.7
k=4	0.420	483.4	0.411	440.9

Table IX compares the maximum noise voltage and skew between the MT and MA fault models for different k's. For k=2, both methods show almost the same maximum noise and delay. The simulation for k=3,4 shows that considerable increase in delay can be stimulated by the MT model compared to the MA model while the peak noise is slightly more.

VI. SUMMARY

We proposed an enhanced boundary scan architecture for testing signal integrity in SoC interconnects. The enhanced architecture is utilized to generate and apply MT-patterns. The proposed MT fault model is a superset of the MA model and is much more capable of testing the capacitive and inductive couplings among interconnects. Our architecture detects integrity loss violations on interconnects based on the widely-used JTAG boundary scan architecture. To do this, additional detector cell, modified scan cells and minor modifications to the TAP controller to handle two new instructions are needed. The advantage of this proposed architecture is that it provides cost effective solution for thorough testing of interconnects using the popular JTAG standard.

ACKNOWLEDGEMENTS

This work has been supported in part by the National Science Foundation CAREER Award #CCR-0130513.

REFERENCES

[1] A. Sinha, S.K. Gupta and M.A. Breuer, "Validation and Test Issues Related to Noise Induced by Parasitic Inductances of VLSI Interconnects," in Proc. *IEEE Trans. on Advanced Packaging*, pp. 329-339, 2002.

[2] X. Bai, S. Dey and J. Rajski, "Self-Test Methodology for At-Speed Test of Crosstalk in Chip Interconnects," in Proc. *Design Automation Conf. (DAC'00)*, pp. 619-624, 2000.

[3] L. Chen, X. Bai and S. Dey, "Testing for Interconnect Crosstalk Defects Using On-Chip Embedded Processor Cores," in Proc. *Design Automation Conf. (DAC'01)*, pp. 317-322, 2001.

[4] A. Attarha and M. Nourani, "Testing interconnects for noise and skew in gigahertz SoCs," in Proc. *Int. Test Conf. (ITC'01)*, pp. 305-314, 2001.

[5] IEEE Standard 1149.1-2001,"*Standard Test Access Port and Boundary-Scan Architecture"*, IEEE Standards Board, 2001.

[6] M. Bushnell, V. Agrawal, *Essentials of Electronics Testing*, Kluwer Publishers, 2000.

[7] M. Cuviello, S. Dey, X. Bai and Y. Zhao, "Fault Modeling and Simulation for Crosstalk in System-on-Chip Interconnects," in Proc. *Intern. Conf. on Computer Aided Design (ICCAD'99)*, pp. 297-303, 1999.

[8] Y. Zhao and S. Dey, "Analysis of Interconnect Crosstalk Defect Coverage of Test Sets," in Proc. *Int. Test Conf. (ITC'00)*, pp. 492-501, 2000.

[9] Nagaraj NS, P. Balsara and C. Cantrell, "Crosstalk Noise Verification in Digital Designs with Interconnect Process Variations," in Proc. *Int. Conf. on VLSI Design*, pp. 365-370, 2001.

[10] W. Chen, S. Gupta and M. Breuer, "Analytic Models for Crosstalk Delay and Pulse Analysis Under Non-Ideal Inputs," in Proc. *Int. Test Conf. (ITC'97)*, pp. 809-818, 1997.

[11] S. Naffziger, "Design Methodologies for Interconnects in GHz+ ICs," Tutorial Lecture in *Int. Solid-State Conf.*, 1999.

[12] Y. Cao, et. al, "Effective On-Chip Inductance Modeling for Multiple Signal Lines and Application to Repeater Insertion," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, no. 6, pp. 799-805, Dec. 2002.

[13] W. Chen, S. Gupta and M. Breuer, "Test Generation for Crosstalk-Induced Delay in Integrated Circuits," in Proc. *Int. Test Conf. (ITC'99)*, pp. 191-200, 1999.

[14] W. Chen, S. Gupta and M. Breuer, "Test Generation in VLSI Circuits for Crosstalk Noise," in Proc. *Intern. Test Conf. (ITC'98)*, pp. 641-650, 1998.

[15] Attarha and M. Nourani, "Test Pattern Generation for signal Integrity Faults on Long Interconnects," in Proc. *VLSI Test Symp. (VTS'02)*, pp. 336-341, 2002.

[16] Y. Zhao, L. Chen and S. Dey, "On-line Testing of Multi-source Noise-induced Errors on the Interconnects and Buses of System-on-Chips," in Proc. *Int. Test Conf. (ITC'02)*, pp. 491-499, 2002.

[17] S. Yang, C. Papachristou, and M. Tabib-Azar, "Improving Bus Test Via I_{DDT} and Boundary Scan," in Proc. *Design Automation Conf. (DAC'01)*, pp. 307-312, 2001.

[18] International Technology Roadmap for Semiconductors 2001 (http://public.itrs.net).

[19] I. Rayane, J. Velasco-Medina and M. Nicolaidis, "A Digital BIST for Operational Amplifiers Embedded in Mixed-Signal Circuits," in *Proc. VLSI Test Symp. (VTS'99)*, pp. 304-310, 1999.

[20] S. Tabatabaei and A. Ivanov, "An Embedded Core for Sub-Picosecond Timing Measurements," in Proc. *Int. Test Conf. (ITC'02)*, pp. 129-137, Oct. 2002.

[21] F. Caignet, S. Delmas-Bendhia and E. Sicard, "The Challenge of Signal Integrity in Deep-Submicrometer CMOS Technology," in Proc. of the IEEE, vol. 89, no. 4, pp. 556-573, April 2001.

[22] M. H. Tehranipour, N. Ahmed and M. Nourani, "Testing SoC Interconnects for Signal Integrity Using Boundary Scan", in Proc. *VLSI Test Symposium (VTS'03)*, pp. 163-168, 2003.

[23] C. Chiang and S. K. Gupta, "BIST TPGs for Faults in Board Level Interconnect via Boundary Scan", in Proc. *VLSI Test Symposium (VTS'97)*, pp. 376-382, 1997.

[24] K. Lofstrom, "Early Capture for Boundary Scan Timing Measurement", in Proc. *Int. Test. Conf. (ITC'96)*, pp. 417-422, 1996.

[25] IEEE P1500 standard, *http://grouper.ieee.org/groups/1500/*.

[26] IEEE 1149.4 standard, *http://grouper.ieee.org/groups/1149/4/*.

[27] L. Whetsel, "Proposal to Simplify Development of a Mixed Signal Test Standard," in Proc. *Int. Test Conf. (ITC'96)*, pp. 400-409, 1996.

[28] IEEE 1149.6 Working Group, *http://grouper.ieee.org/groups/1149/6/*, 2002.

[29] N. Ahmed, M. H. Tehranipour and M. Nourani, "Extending JTAG for Testing Signal Integrity in SoCs", in Proc. *Design, Automation and Test in Europe (DATE'03)*, pp. 218-223, 2003.

[30] M. H. Tehranipour, N. Ahmed and M. Nourani, "Multiple Transition Model and Enhanced Boundary Scan Architecture to Test Interconnects for Signal Integrity", to appear in *Int. Conf. on Computer Design (ICCD'03)*, 2003.

[31] OEA International, Inc., http://www.oea.com.

[32] TI-SPICE3 User's and reference manual, *1994 Texas Instrument Incorporation*, 1994.

[33] Synopsys Design Analyzer, "User Manual for SYNOPSYS Toolset Version 2000.05-1," Synopsys, Inc., 2000.