SoC Test Architecture Design and Optimization Considering Power Supply Noise Effects

Feng Yuan† and Qiang Xu†‡

†CUhk REliable computing laboratory (CURE)
Dept. of Computer Science & Engineering, The Chinese University of Hong Kong
‡CAS-CUHK Shenzhen Institute of Advanced Integration Technology
Email: {fyuan,qxu}@cse.cuhk.edu.hk

Abstract

Excessive power supply noise (PSN) during testing can erroneously cause good chips to fail the manufacturing test, thus leading to unnecessary yield loss. While there are some emerging methodologies such as PSN-aware test generation technique to tackle this problem, they are not readily applicable in modular system-on-a-chip (SoC) testing. This is because: (i). embedded core tests are usually prepared by core providers who are not knowledgeable about the SoC power distribution network; (ii) embedded cores are usually tested in parallel to reduce testing time, but the associated inter-core PSN effects are not considered in existing SoC test architecture design and optimization process. In this paper, we present a fast inter-core PSN estimation method and use it to guide the test scheduling process to solve the PSN-induced SoC test yield loss problem. In addition, upon observing that the PSN effects usually manifest themselves only during the capture phase for scan-tested cores, we introduce novel design for test (DfT) structures into SoC test controller to avoid PSN effects with negligible testing time penalty. Experimental results demonstrate the effectiveness of the proposed solution.

1 Introduction

With the ever shrinking technology feature size enters nanometer era, integrated circuits (ICs) are becoming increasingly sensitive to power supply noise (PSN), i.e., the noise on the power distribution network (PDN), including the inductive noise ($L\frac{di}{dt}$) and IR voltage drop. For example, it was shown in [21] that a 1% voltage change can cause approximately a 4% change in gate delay in today's 90-nm, 0.9-V technology.

At the same time, recent design evaluations have revealed that test patterns in some designs have multiple times of switching activities over the ones in its mission mode [2]. This will cause a large power supply noise in test mode as both IR-drop and $L\frac{di}{dt}$ increase with additional switching activities. Because it is usually unacceptable to oversize the circuit PDN to accomodate manufacturing test, many chips that would function under mission-mode power supply noise may not pass the tests for timing-related defects, thus leading to unnecessary test yield loss (also known as overkill).

While a number of techniques (e.g., pseudofunctional testing [12] and PSN-aware test generation [1, 5, 23]) have been proposed to address the test yield loss issue, this problem is exacerbated in core-based system-on-a-chip (SoC) testing where the above techniques are not readily applicable. First of all, embedded core tests are usually prepared by core providers who are not knowledgeable about the SoC power distribution network. Consequently, tests that lead to excessive PSN might be generated. In addition, many today's SoC designs adopt the approach of "just-enough" energy to deliver the required functions with minimum power consumption. The accordingly designed power distribution network is thus more sensitive to power supply noise. When we test multiple embedded cores in parallel to reduce testing time while these cores do not operate together in functional mode, we have a more serious concern: switching activities in one core under test (CUT) could cause excessive voltage droop in another CUT and thus lead to false test rejects.

Existing work in power-constrained SoC test architecture design and optimization can alleviate the aforementioned PSN effects, but cannot guarantee to solve this problem. This is because, the objective of the above technique is to reduce the total test power dissipation of the entire chip, which is usually not effective to reduce highly localized switching activities that cause excessive supply voltage drop.

We propose novel solutions for the above problem in this paper. The main contributions include:

- we present a fast *inter-core PSN estimation method* that takes the SoC layout, power distribution network and CUT switching activities into account during the estimation process.
- we present a *PSN-constrained test scheduling algorithm* that is adapted from an existing technique guided by our inter-core PSN analysis results.
- we propose a *novel PSN-aware SoC test controller* that is able to mitigate PSN effects for scan-tested cores with negligible testing time penalty.

To the best of our knowledge, this is *the first work* that explicitly takes inter-core PSN effects into consideration during the SoC test architecture design and optimization process.

1-4244-4203-0/08/$20.00 ©2008 IEEE

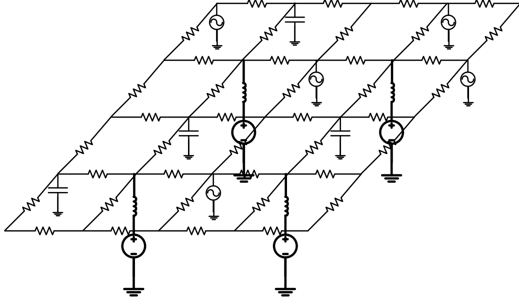


Figure 1. RLC Model for Power Distribution Network

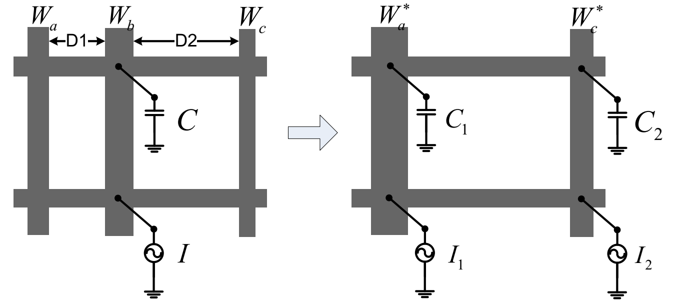


Figure 2. Power Network Reduction [25]

The remainder of this paper is organized as follows: Section 2 reviews related work and motivates this work. In Section 3, we formulate the problem addressed in this paper. The inter-core power supply noise estimation method is then presented in Section 4 and we derive a PSN-constrained test scheduling algorithm based on it. Next, Section 5 proposes a novel PSN-aware SoC test controller and the associated test scheduling algorithm. Experimental results are presented in Section 6. Finally, Section 7 concludes this paper.

2 Preliminaries and Motivation

2.1 PSN Analysis

With the increasing circuit density and functional frequency, today's designs have shorter rise/fall transition time and higher instantaneous current (with more simultaneous switching activities). Both cause more power supply noise to the circuit. The reduced device supply voltage then inevitably leads to performance degradation and signal integrity loss. Therefore, PSN effects have been a major concern for the industry nowadays.

ICs' power distribution network can be modeled as a RCL network, as depicted in Fig. 1. Since the parasitic inductance of on-chip metal power rail is much smaller than the package-level inductance, for the sake of simplicity, the PDN is typically assumed to be a pure resistance network. Each power pin, on the other hand, is modeled with a voltage source connecting to a lumped inductor and a resistor. The exact locations of the power pins depend on the chip package, and can be at the border of the package (for wire-bonding) or at the package bottom (for flip-chip bonding). Moreover, decoupling capacitors are often inserted to alleviate PSN effects and they are modeled as a lumped capacitor between PDN and ground. Finally, switching devices are assumed to be ideal current sources.

Since the power network of today's SoC devices usually consist of millions of nodes, exactly solving the above RLC network is extremely time-consuming, if not impossible. Various power network reduction techniques were proposed to estimate PSN effects with acceptable accuracy. Chiprout [3] presented a fast partition-based power grid analysis technique. [9, 25] exploited multigrid analysis for the same problem. Recently, [16] proposed to use random walk for PSN analysis and achieved satisfactory results. A divide-and-conquer approach was proposed in [29], wherein hierarchical structure were utilized to conduct power grid reduction.

We briefly introduce the multigrid-based method used in [25] here as this method is exploited in this work. The basic idea in [25] is to first solve the RLC network at a coarse-grained level; and then map it back to the fine-grained PDN. The coarse-grained reduced network is obtained by skipping every other row (or column) in the original network. The switching current I on the skipped power rail is assumed to be distributed to its neighbors as follows:

$$I_1 = I \times \frac{D2}{D1+D2} \quad (1)$$

$$I_2 = I \times \frac{D1}{D1+D2} \quad (2)$$

where D1 and D2 are the distances between the neighboring power rail (as shown in Fig. 2). A few iterations of the above steps using classical iterative solver can lead to acceptable accuracy for PSN estimation.

2.2 PSN-Aware Test Generation

At-speed delay testing is widely used in the industry nowadays to detect timing-related defects. Since power supply noise has a significant impact on the timing behavior of the circuit, it is necessary to integrate PSN effects in delay test pattern generation.

Early work in this domain (e.g., [10, 15]) tries to sensitize critical paths with high power supply noise when generating test patterns, in order to avoid test escapes as much as possible. As can be observed in [17, 20], however, because switching activities in test mode is usually much higher than the ones in mission mode, the above strategy can cause a considerable number of false rejects and result in a large test yield loss.

A number of delay test generation techniques were proposed to accommodate the above problem. In [18, 26], the authors try to decrease the switching activities during the scan capture phase by filling the don't care bits in the test cube in such way that the Hamming distance between two successive delay vectors are minimized. [23] proposed an "effective region" concept and used it to guide the delay test pattern generation process. The authors of [1] proposed another PSN-aware delay test pattern generation method by measuring the average power of these at-speed test patterns.

2.3 SoC Test Architecture Design and Optimization

Modular testing using dedicated bus-based test access mechanisms (TAMs), e.g., Test Bus [22] or TestRail [6], is the most popular test strategy for large SoC devices used by the industry, because this methodology facilitates the reuse of pre-computed tests for embedded cores and reduces the complexity of SoC test problem in a "divide and conquer" manner [28]. In modular testing, embedded cores are isolated from their surrounding logic using test wrappers, while TAMs are designed to transport test data between the input/output pins of the SoC and the cores under test (CUTs). An on-chip test controller generates the necessary test wrapper control signals and controls the entire test process according to the test schedules pre-assigned by designers.

The SoC test architecture optimization problem refers to the problem that, for an SoC with specified parameters for its core tests, we are to design a test architecture and a test schedule to minimize the test cost, which must account for the test application time and the amount of on-chip DfT resources (both logic and routing). This problem has been subject to extensive research and various test scheduling algorithms were proposed in the literature [28], including a number of power-constrained test scheduling techniques (e.g., [7, 8, 11, 27]), wherein the main objective is to schedule core tests for testing time reduction under the condition that the total power consumption of those cores under test is not greater than a given power constraint at any time. Consequently, the total switching activities of the SoC device are under control with the above manner. This is helpful for power supply noise reduction and hence could implicitly alleviate the PSN-induced overkill problem.

2.4 Motivation

Because whether switching activities in one core lead to significant voltage drop in another core depends on the SoC power distribution network and more importantly the physical positions of these two cores, without considering the above issues explicitly, existing power-constrained test architecture optimization strategies cannot effectively solve the PSN-induced overkill problem. This has motivated us to introduce inter-core PSN analysis into the SoC test architecture optimization process in this work.

Generally speaking, if a core is scan tested, the excessive switching activities in other cores will have higher chance to cause test failures if these transitions occur in its at-speed capture phase. The supply voltage drop induced by other cores will not invalidate the shift process because the shift frequency is usually much slower and hence can tolerate the extra delay resulted from supply voltage drop. The above observation has motivated us to design a novel PSN-aware SoC test controller to stop the transitions of other cores when a scan-tested core is in its capture phase. By doing so, PSN-safe test can be achieved with a negligible testing time penalty.

It should be noted that, however, many embedded cores are not scan-tested (e.g., built-in self-test (BIST) for memory cores or functional test for processor cores), and hence the new PSN-aware SoC test controller itself cannot solve the PSN-induced test yield loss problem completely. Rather, we should take the inter-core PSN effects into account to optimize the SoC test architecture containing the new controller, so that a more cost-effective solution can be achieved.

3 Problem Formulation

As discussed earlier, when multiple embedded cores are tested concurrently, their switching activities will cause voltage drops on each other. In this work, we take this *inter-core PSN* effects into account during the SoC test architecture design and optimization process. The problem addressed in this paper can be formulated as follows:

Problem: Given

- the SoC test parameters, including the number of cores N_c, the power constraint P_{max} and the given TAM width W_{TAM};
- the test parameters for every core. For each $core_i$ ($1 \leq i \leq N_c$), given sc_i internal scan chains, $l_{i,1}, \ldots, l_{i,sc_i}$ internal scan chain lengths, in_i wrapper input cells, out_i wrapper output cells, bi_i wrapper bidirectional cells, the supply voltage Vdd_i, and t_i tests (e.g., stuck-at test or delay test). For each test j of core i ($1 \leq j \leq t_i$), given the test pattern count $n_{i,j}$, the maximum switching activity ratio $sa_{i,j}$ (i.e., the percentage of switching devices), the test power $p_{i,j}$, and the maximum tolerable voltage drop $VD_{i,j}$;
- the layout information of every core. For each $core_i$ ($1 \leq i \leq N_c$), given the coordinates of its left lower corner (x_i, y_i), width w_i and height h_i (the size of core i is thus $s_i = w_i \times h_i$);
- the SoC power distribution network parameters, including the placement of power pads, the PDN topology, the sheet resistance of power rail, the package inductance, the placement and capacitance of decoupling capacitors;
- the test current information for every core. For each $core_i$ ($1 \leq i \leq N_c$), the maximum direct current I_i and the maximum current variation ΔI_i for a switching device;

Design a test architecture and a test schedule for the SoC such that the total SoC testing time is minimized while satisfying the following constraints:

- the total utilized TAM width at any time does not exceed W_{TAM};
- the total test power consumption at any time does not exceed P_{max};
- the PSN-induced voltage drop for any core i under test j does not exceed $VD_{i,j}$;

4 PSN-Constrained Test Architecture Optimization

4.1 Inter-Core PSN Estimation

Since we are usually not knowledgeable about the internal structures of embedded cores in an SoC and hence the actual switching devices when applying a particular test pattern, it is impossible to determine the exact voltage drop for every device inside these cores. During our estimation process for inter-core power supply noise, we assume that all devices and power pads are attached to the nearest crossing point (denoted as *power node*) on the PDN, and we only consider the voltage drop at power nodes (in total n of it). After reducing the PDN size with the techniques presented in [25], a modified nodal analysis method is utilized for inter-core PSN estimation:

$$\mathbf{G}\mathbf{x}(t)+\mathbf{C}\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t}=\mathbf{b}(t) \tag{3}$$

$\mathbf{G}=[g_{ij}]_{n\times n}$ is conductance matrix, in which g_{ii} is the sum of all conductances connected to node i while $g_{ij}(i\neq j)$ is the negative conductance between node i and node j. The above values can be easily obtained from the sheet resistances and the sizes of power rails. $\mathbf{C}=[c_{ij}]_{n\times n}$ is the matrix derived from capacitive and inductive elements. In this matrix, c_{ii} include the capacitance and inductance attached to this node while $\forall i\neq j: c_{ij}=0$. $\mathbf{b}(t)$ is a column vector corresponding to the ideal voltage source and time-varying current source for every node. Finally, $\mathbf{x}(t)$ is a vector that represents the to-be-estimated voltage for every node.

Equation (3) can be reduced to a set of linear equations [19]:

$$(\mathbf{G}+\frac{\mathbf{C}}{h})\mathbf{x}(t)=\mathbf{b}(t)+\frac{\mathbf{C}}{h}\cdot\mathbf{x}(t-h) \tag{4}$$

$\mathbf{x}(t)$ can then be exactly solved by using Backward Eular technique with a small fixed time-step h, if the time-varying current waveform at every power node is given. Unfortunately, the above information is usually not available because it is impossible for us to simulate test patterns without knowing the embedded core structure.

In this work, for a core under test, assuming its switching ratio and switching device distribution are given by IP vendor (e.g., 25% switching ratio and evenly distributed to every switching device), we consider the case where the maximum direct current (denoted as $\mathbf{b}$) and the maximum current variance (denoted as $\Delta\mathbf{b}$) happen at the same time for all power nodes on this core. By doing so, $\mathbf{b}(t)$ is reduced to be $\mathbf{b}+h\Delta\mathbf{b}$ and we do not require to know the time-varying current waveform. We then use a two-step simulation method[1] to solve Equation (4) as follows:

1. in this step, we only consider the linear elements in PDN, and calculate $\mathbf{x_d}$ (containing direct voltage only) by solving linear equation $\mathbf{G}\cdot\mathbf{x_d}=\mathbf{b}$;
2. in this step, we substitute $\mathbf{x_d}$ obtained from previous step for $\mathbf{x}(t-h)$ and take nonlinear elements (i.e., inductances and capacitances) into account, and solve the equation $(\mathbf{G}+\frac{\mathbf{C}}{h})\mathbf{x}=\mathbf{b}+h\Delta\mathbf{b}+\frac{\mathbf{C}}{h}\mathbf{x_d}$;

[1]This fast approximation method may over-estimate PSN-induced voltage drop, but it guarantees to generate safe tests.

After obtaining $\mathbf{x}$ using the above method, we are able to acquire the inter-core PSN effects for every pair of core tests. For the sake of simplicity, we assume a core contains at most one test (e.g., delay test) that is affected by other core tests (the maximum tolerable voltage drop $VD_{i,j}$ in Section 3 can then be simplified as VD_i). We can thus obtain a matrix $\mathbf{PSN}=[psn_{ij}]$, in which $psn_{ii}=0$ while $psn_{ij}(i\neq j)$ indicates the maximal power node voltage drop on $core_j$ when $core_i$ is concurrently tested.

4.2 PSN-Constrained Test Scheduling

With the inter-core PSN estimation framework discussed earlier, we introduce a PSN-constrained test scheduler by adapting an existing algorithm presented in [8], briefly described as follows (please refer to [8] for details):

$S_{unscheduled}$, S_{CUT}, and $S_{finished}$ are defined as the set of cores that have not been scheduled to test yet, the set of cores that are currently under test, and the set of cores that have finished their tests, respectively, for a particular schedule time. An initialization procedure is utilized to assign every core a preferred TAM width. All cores are sorted in a non-increasing order based on their testing times with the preferred TAM widths and put into $S_{unscheduled}$. The test scheduler then iteratively removes a core from $S_{unscheduled}$, puts it into S_{CUT} and allocates TAM resources to it. The selection criteria include the order in $S_{unscheduled}$, whether scheduling the core results in any constraint violation (e.g., power constraint), whether the core can be tested with preferred TAM width, etc. Once a core finishes its test, it will be removed out of S_{CUT} and put into $S_{finished}$. The scheduling process ends when $S_{finished}$ contains all embedded cores of the SoC.

To import the PSN-constraints into the above scheduling algorithm, we need to calculate the cumulative PSN-induced voltage drop for a core under test. Let us use S^T to represent a set of concurrently tested cores during a test session T (the concurrent core tests remain unchanged in a test session), for $core_i\in S^T$, its cumulative voltage drop can be calculated as follows:

$$vd_i=\sum_{j=1,core_j\in S^T}^{|N_c|} psn_{ji} \tag{5}$$

S^T is called a *PSN-compatible* core set if $vd_i\leq VD_i$ for any $core_i\in S^T$. Otherwise, it is a *PSN-incompatible* core set. Therefore, by keeping S_{CUT} to be PSN-compatible all the time during the scheduling process, we are able to obtain a PSN-constrained test schedule for the SoC.

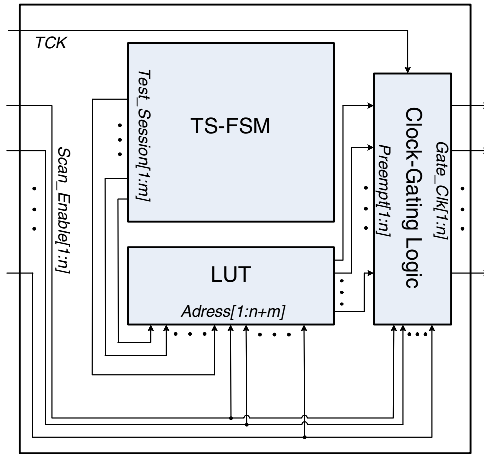


Figure 3. Block Diagram of the Proposed PSN-Aware Controller

$Scan_Enable[1:3]$	$Preempt[1]$	$Preempt[2]$	$Preempt[3]$
011	0	1	0
101	1	0	1
110	0	0	0
111	0	0	0

Table 1. LUT Values for a Test Session.

5 PSN-Aware SoC Test Architecture Design and Optimization

5.1 PSN-Aware Test Controller

Full scan is the mainstream DfT technique utilized by the industry. Scan-based at-speed testing usually involves a long period of shift phase and a short at-speed capture phase. As the shift frequency is typically kept quite low to reduce test power consumption, test patterns are able to shift correctly with very low supply voltage. Therefore, generally speaking, the PSN-induced voltage drop only affects the capture phase for scan-tested cores. Let us use S^T_{scan} and $S^T_{non-scan}$ to represent the set of scan-tested cores and the set of cores that are not scan-tested (e.g., BISTed), in a concurrently tested core set S^T during a test session T, and $S^T = S^T_{scan} \cup S^T_{non-scan}$. From the above, for any $core_i \in S^T_{scan}$, as long as the PSN-induced voltage drop during its capture phase is smaller than VD_i, it is safe to apply this test. Therefore we can relax our prior definition for $PSN-compatibility$. That is, a concurrently tested core set S^T is $PSN-compatible$

- if $vd_i \leq VD_i$ for any $core_i \in S^T_{non-scan}$ during the entire test session T;
- if $vd_i \leq VD_i$ for any $core_i \in S^T_{scan}$ during the scan capture phase in T;

Consider a PSN-incompatible core set S^T containing one at-speed scan-tested $core_i \in S^T_{scan}$ that has PSN violation (i.e., $vd_i > VD_i$), if we are able to suppress the switching activities in $S^T \setminus \{core_i\}$ (i.e., other cores in S^T) to lower vd_i so that $vd_i \leq VD_i$ during its capture phase, we can make S^T to be PSN-compatible. By doing so, $core_i$ does not need to be scheduled at other time, thus reducing the total SoC testing time. The relevant question is then: can we reduce switching activities in $S^T \setminus \{core_i\}$? The answer is positive provided that the tests of some cores in $S^T \setminus \{core_i\}$ are *preemptable* (i.e., can be interrupted and resumed later).

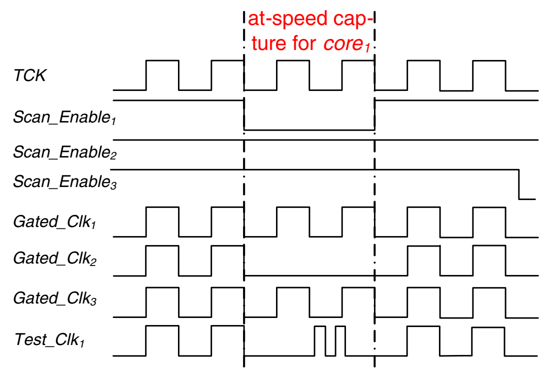


Figure 4. Timing Diagram with the PSN-Aware Controller

As scan tests are preemptable, we propose a novel PSN-aware SoC test controller to achieve the aforementioned functionality for such cores. That is, if a $core_i \in S^T_{scan}$ has PSN violations, during its capture cycles, we hold back the operations of other cores in S^T_{scan} (if any) by gating their test clock signals. The testing time of these cores in $S^T_{scan} \setminus \{core_i\}$ will be slightly increased due to the pause of their shift operations (e.g., two clock cycles per test pattern in test session T). However, this is usually negligible when compared to the cost of scheduling $core_i$ to a later time.

The block diagram of the proposed PSN-aware SoC test controller is shown in Fig. 3. The *TS-FSM* is a finite state machine (FSM) that holds the test schedule information and controls the test session switchings. Within a test session T, for any scanned $core_i \in S^T$, we need to know whether we should preempt the shift operations of cores in $S^T \setminus \{core_i\}$ during its at-speed capture phase. We put this information in a look-up table (LUT) and route the scan enable signals to the test controller to serve as address of the LUT. The values stored in this LUT are the *Preempt* signals used for gating the test clock signal TCK, which are then used to generate the shift clock signals $Gated_Clk$ for the scanned cores in current test session (concurrent core tests remain unchanged within each test session).

Consider an example test session that contains at-speed tests of scanned-cores $core_1$, $core_2$ and $core_3$. The values stored in the LUT is shown in Table 1. We can see when $Scan_Enable[1:3] = `011'$, $Preempt[2] = `1'$, which means that $core_2$'s shift operations need to be preempted so that $core_1$'s at-speed capture can be applied without PSN violations. At the same time, $Preempt[3] = `0'$ implies that the shift operations of $core_3$ does not lead to significant PSN effects on $core_1$'s capture and hence can be continued without interruption. The timing diagram for the above case when $core_1$ is in its at-speed capture phase (i.e., $Scan_Enable[1:3] = `011'$) is depicted in Fig. 4, in which the shift clock signal for $core_2$ is gated (see $Gated_Clk_2$).

***combined* algorithm**

scheduler()

1. initialization(); — Pre-calculate preferred TAM width to every core and get the corresponding preferred testing time; set $S_{unscheduled}$ to contain all cores and sort it in non-increasing order according to preferred testing time; set $S_{finished} = \emptyset$; $S_{CUT} = \emptyset$; $S_{non-scan} = \emptyset$; $S_{scan} = \emptyset$); set $W = W_{TAM}$; $Current_Time = 0$;
2. **while**($S_{unscheduled}$ or S_{CUT} is not empty) {
3.   **for**($i = 0; i \leq |S_{unscheduled}|; i++$) {
4.   **if**($S_{unscheduled}(i)$ is a non-scanned core) {
5.   **if**(validation_checking($S_{unscheduled}(i)$)) { — determine whether we should schedule this core;
6.   move $S_{unscheduled}(i)$ from $S_{unscheduled}$ to $S_{non-scan}$;
7. $Start_Time(S_{unscheduled}(i)) = Current_Time$;}}
8.   **else** {
9.   **if**($S_{CUT} \cup \{S_{unscheduled}(i)\}$ is *PSN-compatible*) { — If $S_{unscheduled}(i)$ does not cause PSN violation, schedule it as in [8];
10.   **if**($S_{unscheduled}(i)$'s TAM requirement can be satisfied) {
11.   assign W_i TAM to $S_{unscheduled}(i)$;
12. $W = W - W_i$;}}
13.   **if**($W \neq 0$) {
14.   reassign residual W to the cores start at $Current_Time$ until no core's testing time can be reduced;}
15.   update();}}}

validation_checking($S_{unscheduled}(i)$)

1. **if**($S_{CUT} \cup S_{unscheduled}(i)$ is *PSN-incompatible*)
2.   **return false**;
3. $violated_core_number = 0$; — $violated_core_number$ is the number of unscheduled non-scanned cores that cannot be scheduled with $S_{unscheduled}(i)$;
4. **for**($j = 0; j < |S_{unscheduled}|, j \neq i; j++$) {
5.   **if**($S_{unscheduled}(i) \cup S_{non-scan} \cup S_{unscheduled}(j)$ is *PSN-incompatible*) {
6. $violated_core_number++$;}}
7. **if**($violated_core_number > |S_{unscheduled}|/2$)
8.   **return false**;
9. **return true**;

update()

1. calculate End_Time for every core starts its schedule at $Current_Time$;
2. set $Next_Time$ to be the first core end time in S_{CUT};
3. if(S_{CUT} is *PSN-incompatible* without the PSN-aware test controller) {
4.   stop those scanned cores during the capture cycles of PSN-violated cores from $Current_Time$ to $Next_Time$;
5.   update End_Time for the above scanned cores; update $Next_Time$;}
6. update $S_{unscheduled}$, $S_{finished}$, S_{CUT}, $S_{non-scan}$, S_{scan};
7. $Current_Time = Next_Time$; update $Next_Time$;

Figure 5. Test Scheduling Algorithm *Combined* for SoCs Containing PSN-Aware Test Controller.

5.2 Combined Test Scheduling

As discussed above, the newly-introduced test controller is able to alleviate PSN effects caused by scan-tested cores. If all embedded cores in the SoC are scan-tested, essentially we can ignore PSN constraints and apply any algorithm to schedule the SoC test (e.g., [8]), by completing the obtained schedule with a post-processing step to stop certain cores during the PSN-violated capture cycles. In practice, however, many cores might be BISTed (e.g., memory cores) or functionally-tested (e.g., processors). The proposed PSN-aware test controller alone thus cannot solve the PSN-induced overkill problem completely. For such cases, we should combine this method with effective PSN-constrained test scheduling technique to achieve a cost-effective solution, as illustrated in this section.

In our scheduling algorithm, we consider the case for SoC containing both scanned cores and non-scanned cores. In addition, we assume the scanned cores transfer test data to/from external tester through TAMs, while the non-scanned cores have separate test sources/sinks on-chip (e.g., BIST engine) and do not utilize TAMs for test data transfer (e.g., memory BIST). Again, we adapt the constraint-driven test scheduling algorithm in [8] in our test scheduler, denoted as *combined* algorithm as shown in Fig. 5.

Before describing the details of the scheduling process, let us introduce the terms used in the algorithm first. $Current_Time$ is the current schedule time. $S_{unscheduled}$ and S_{CUT} denote the set of cores that have not been scheduled to test yet till $Current_Time$ and the set of cores that are currently under test at $Current_Time$, respectively. Within S_{CUT}, we use S_{scan} and $S_{non-scan}$ to denote the set of scanned cores and non-

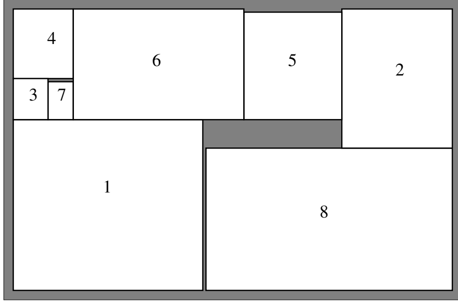


Figure 6. An Example Layout for h953

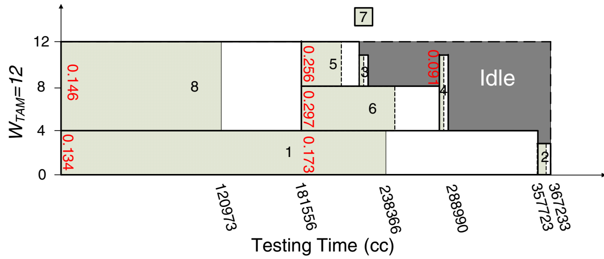


Figure 7. PSN-Induced Voltage Drop for h953 with Power-Constrained Test Scheduler [8]

scanned cores. For a $core_i$ that has been scheduled to test, $Start_Time(i)$ and $End_Time(i)$ denotes its schedule start time and schedule end time, respectively. Finally, $Next_Time$ is the earliest End_Time for cores in S_{CUT}.

The basic scheduling procedure is similar to the one in [8], as can be observed in Fig. 5. There are, however, also some significant differences. Recall that the new SoC test controller is able to eliminate the PSN effects induced by scanned cores, but cannot suppress the switching activities in non-scanned cores. For non-scanned cores, since they do not use TAMs to transfer test data, they are mainly constrained by inter-core PSN constraints and power constraints. If many such cores are scheduled concurrently, the associated large switching activities may lead to significant PSN effects to those scanned cores and make them impossible to schedule at the same time. This will lead to under-use of the precious TAM resources and thus is not a preferred strategy. To tackle this problem, we introduce a procedure namely *validation_checking* into the scheduling process. Whenever a non-scanned core is selected from $S_{unscheduled}$, we apply this procedure to make sure that adding this core to S_{CUT} would not violate the PSN constraints for most of the unscheduled cores. In addition, the *combined* algorithm is also responsible for inserting idle cycles for those scanned cores that lead to PSN violations without suppressing their activities. This is conducted in the *update* procedure, as shown in Table 5.

6 Experimental Results

6.1 Experimental Setup

To examine the effectiveness of the proposed solution, we present experimental results for three ITC'02 benchmark SoCs [13]: h695, p22810, and p34392. As these benchmark circuits do not contain all the necessary information used to calculate inter-core power supply noise effects (e.g., the circuit layout). We generate them as follows.

For cores containing scan chains (typically logical cores), we use the total number of inputs/outputs (I/Os) and internal flip-flops to estimate their relative sizes $s_{relative}$. For cores that do not include any scan chains (e.g., memory cores), we use the number of test patterns to estimate their $s_{relative}$. The silicon area of the core is then set to be: $s = s_{relative} \times 10^{-5}\ cm^2$. Assuming the aspect ratios of these cores range from 0.5 to 2, we use an academic floorplanning tool [14] to generate a random SoC layout. An example floorplan for SoC h695 is shown in Fig. 6 with total silicon area $0.05cm^2$.

For the on-chip power distribution network, a regular mesh-based PDN is assumed to provide 1V supply voltage to all devices, with $3\mu m$-wide power line, $60\mu m$-wide pitch, and $0.1ohm$ sheet resistance [4]. A number of $1.5pf$ decoupling capacitors are distributed around cores. The total number of gates in a core is assumed to be $20s_{relative}$. For each core, its switching gates are assumed to be evenly distributed physically and the worst switching activities for all core tests are set as 20% of the total gates. The current source that drives a switching gate is assume to have $150\mu A$ peak current and $7A/s$ peak transient current variance, as in [24].

As PSN effects mainly manifest themselves during at-speed testing (e.g., delay test and memory BIST). We assume the original test in the SoC benchmark for those cores with scan chains are stuck-at tests that are not affected by PSN effects, and we introduce a separate at-speed delay test for every such core, with twice number of test patterns for the corresponding stuck-at test. These scan tests are preemptable. For the test for those cores without internal scan chains, we assume they are non-preemptable at-speed BIST tests applied with dedicated BIST engine and their testing times equal to their test pattern counts. The maximal tolerable power supply noise VD_i is set as 0.1V for at-speed test of every $core_i$.

Finally, power constraints are considered in our experiments. For benchmark h953, we use the values given in the benchmark; for p22810 and p34392, the power consumption information are not provided, and we estimate the power consumed by a core to be same as its relative size $s_{relative}$ while the total system power constraint is set to be 20% of the total power of all cores.

6.2 Results and Discussions

First of all, let us examine whether the power-constrained test scheduling technique [8] is able to eliminate the PSN-induced overkill problem. Fig. 7 presents such a test schedule for SoC h953 when the given TAM width $W_{TAM} = 12$ (see Fig. 6 for its layout). It can be easily observed that the voltage drops more than 0.1V for several cores even though the total system power constraint is satisfied, wherein $core_6$ suffers from the most voltage drop (~ 0.3V). Note $core_7$ is a non-scanned core and does not occupy any TAM resources for test data transfer in our experiment (see Fig. 7). This is because the inter-core PSN is strongly related to the physi-

W_{TAM}	$T_{baseline}$	T_{psn}	ΔT_{psn}	$T_{combined}$	$\Delta T_{combined}$
4	885625	885625	0	885625	0
8	703557	658506	-6.4%	553105	-21.4%
12	700301	548789	-21.6%	368586	-47.6%
16	595947	541461	-9.1%	368586	-38.2%
20	597830	541461	-9.4%	368586	-38.3%
24	598694	541461	-9.6%	368586	-38.4%
28	590614	541461	-8.3%	368586	-37.6%
32	591702	541461	-8.5%	368586	-37.7%

$\Delta T_{psn} = \frac{T_{psn} - T_{baseline}}{T_{baseline}}$; $\Delta T_{combined} = \frac{T_{combined} - T_{baseline}}{T_{baseline}}$

Table 2. Testing Time Comparison for h953.

W_{TAM}	$T_{baseline}$	T_{psn}	ΔT_{psn}	$T_{combined}$	$\Delta T_{combined}$
8	2703296	2703296	0	2703296	0
16	1322521	1322521	0	1319396	-0.0%
24	1106047	1048980	-5.2%	1008518	-8.8%
32	939902	789401	-16.0%	758139	-19.3%
40	906309	625858	-30.9%	656537	-27.6%
48	839134	625858	-25.4%	543390	-35.2%
56	776121	625858	-19.4%	543390	-30.0%
64	789606	625858	-20.7%	543390	-31.1%

Table 3. Testing Time Comparison for p22810.

W_{TAM}	$T_{baseline}$	T_{psn}	ΔT_{psn}	$T_{combined}$	$\Delta T_{combined}$
8	5594856	5594856	0	5594856	0
16	3317911	3305272	-0.4%	3305272	-0.4%
24	3142968	2596804	-17.4%	1704412	-40.0%
32	2899986	2364804	-18.5%	1657247	-41.2%
40	3052222	2364804	-22.5%	1657247	-45.7%
48	3052222	2364804	-22.5%	1657247	-45.7%
56	3052222	2364804	-22.5%	1657247	-45.7%
64	3052222	2364804	-22.5%	1657247	-45.7%

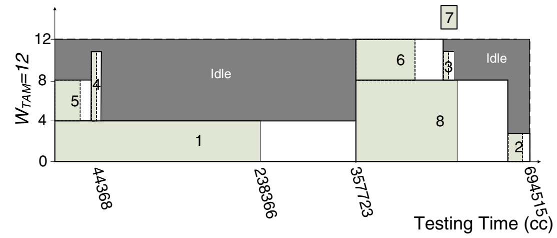
Table 4. Testing Time Comparison for p34392.

cal position of on-chip cores. $core_6$ is scheduled to be tested with its neighboring cores $core_1$ and $core_5$, and hence suffers more inter-core power supply noise. From this example, we can see that power-constrained test schedule may lead to PSN-induced overkill problem.

To demonstrate the effectiveness of the proposed solution, we compare with a baseline solution (denoted as *Baseline*) by post-processing the test schedule obtained from power-constrained test scheduling technique. That is, we identify those concurrent tests that violate PSN constraint and postpone some of the tests to avoid violations. We use *PSN-Constrained* and *Combined* to represent the test schedules obtained from the proposed PSN-constrained test scheduling technique (see Section 4) and the one acquired from the combined algorithm with the help of the power-aware SoC test controller (see Fig. 5).

Tables 2, 3 and 4 present testing times for the three benchmark SoCs: h953, p22810, and p34392, wherein $T_{baseline}$, T_{psn} and $T_{combined}$ represent the testing time obtained from the baseline solution, the PSN-constrained test scheduling algorithm presented in Section 4.2 and the *combined* algorithm presented in Section 5.2, respectively. For h953 we vary the TAM width W_{TAM} from 4 to 32, while for p22810 and p34392 we vary W_{TAM} from 8 to 64.

(a) Scheduling result for Baseline Case



(b) Scheduling result for PSN-Constrained Case

Figure 8. Test Schedule Comparison for h953.

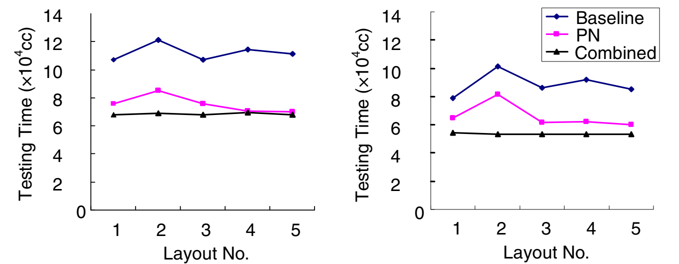


Figure 9. Testing Time Comparison for p22810 with Different Layouts.

From these tables, we can observe that the proposed PSN-constrained scheduler has better performance than the baseline case for all three benchmark SoCs. Fig. 8 demonstrates the test schedules for h953 obtained from the two algorithms. Apparently, by taking inter-core PSN effects into account during the test scheduling process, we can effectively test multiple cores in parallel to reduce testing time without violating a given PSN constraint.

In addition, from these tables we can also find that the more available TAM resources, the more benefits we can gain from the *combined* scheduler. This is because, with the increase of TAM width, the test parallelism is mainly limited by the power supply noise constraint. With the help of the PSN-aware SoC test controller, the *combined* scheduler suffers less power supply noise and hence is able to better exploit the benefits from more TAM resources when compared to the other two solutions.

In benchmark p34392, the testing time of *Combined* and *PSN_Constrained* scheduler does not decrease as the increasing of W_{TAM} after it exceeds 32. That is mainly because this particular SoC contains a very large core and it becomes the scheduling bottleneck once W_{TAM} exceeds the value that can results in its testing time reduction. The same situation applies to h953 once W_{TAM} exceeds 12.

Finally, we show the impact of the SoC physical layout on testing times when considering inter-core PSN effects, as shown in Fig. 9. Five different layouts for benchmark p22810 are randomly generated. For the baseline and the PSN-constrained test scheduling cases, the testing times vary significantly with different layout. This is because the inter-core power supply noise is strongly dependent on the core positions. We can also see that proposed PSN-constrained test scheduler always obtains better results than the baseline case, which proves the effectiveness of this algorithm. It can be also observed that the testing times do not very much when we use the proposed *combined* algorithm. This is expected because p22810 is mainly composed of scanned cores, where their inter-core PSN effects can be eliminated by the PSN-aware test controller, and thus we can use almost the same test schedule for all the layouts.

7 Conclusion and Future Work

In this paper, we present a SoC test architecture design and optimization framework that takes inter-core power supply noise effects into account, to address the ever-increasing PSN-induced overkill problem. In particular, we propose a novel PSN-aware test controller design and a PSN-constrained test scheduling technique. Experimental results on a revised version of ITC'02 benchmark SoC demonstrate the effectiveness of the proposed solution.

In this work, we only consider the inter-core PSN effects among cores that are currently under test. With the shrinking technology feature size, however, even if a core is not currently under test, its leakage current may affect the power supply voltages of its neighboring cores, especially when its temperature is high (e.g., because it just finishes its test). We plan to take this into consideration in our future work. In addition, we would like to make the proposed PSN-aware test controller to be reconfigurable so that it can be flexible enough to support the change of test schedules at a later stage.

8 Acknowledgement

This work was supported in part by the Hong Kong SAR RGC Earmarked Research Grant 417406 and 417807, and in part by the National High Technology Research and Development Program of China (863 program) under grant no. 2007AA01Z109.

References

[1] N. Ahmed, M. Tehranipoor, and V. Jayaram. Supply Voltage Noise Aware ATPG for Transition Delay Faults. In *Proc. VTS*, pp. 179–186, 2007.

[2] C. Shi and R. Kapur. How Power Aware Test Improves Reliability and Yield. EE Times, Sept. 15, 2004.

[3] E. Chiprout. Fast flip-chip power grid analysis via locality and grid shells. In *Proc. ICCAD*, pp. 485–488, 2004.

[4] J. Choi, et al. Modeling of Power Supply Noise in Large Chips Using the Circuit-Based Finite-Difference Time-Domain Method. *IEEE Transactions on Electronmagnetic Compatibility*, 47(3):424–439, August 2005.

[5] V. R. Devanathan, C. P. Ravikumar, and V. Kamakoti. Variation-Tolerant, Power-Safe Patttern Generation. *IEEE Design & Test of Computers*, 24(4):374–384, July-Aug. 2007.

[6] S. Goel and E. Marinissen. Effective and efficient test architecture design for SOCs. In *Proc. ITC*, pp. 529–538, 2002.

[7] Y. Huang et al. Resource Allocation and Test Scheduling for Concurrent Test of Core-Based SOC Design. In *Proc. ATS*, pp. 265–270, 2001.

[8] V. Iyengar, K. Chakrabarty, and E. J. Marinissen. Integrated Wrapper/TAM Co-Optimization, Constraint-Driven Test Scheduling, and Tester Data Volume Reduction for SOCs. In *Proc. DAC*, pp. 685–690, 2002.

[9] J. Kozhaya, S. Nassif, and F. Najm. A Multigrid-Like Technique for Power Grid Analysis. *IEEE Transactions on Computer-Aided Design*, 21(10):1148–1160, Octorber 2002.

[10] A. Krstic, Y.-M. Jiang, and K. T. Cheng. Pattern Generation for Delay Testing and Dynamic Timing Analysis Considering Power-Supply Noise Effects. *IEEE Transactions on Computer-Aided Design*, 20(3):416–425, March 2001.

[11] E. Larsson and Z. Peng. A Reconfigurable Power-Conscious Core Wrapper and its Application to SOC Test Scheduling. In *Proc. ITC*, pp. 1135–1144, 2003.

[12] Y.-C. Lin, F. Lu, and K. Cheng. Pseudofunctional Testing. *IEEE Transactions on Computer-Aided Design*, 25(8):1535–1546, August 2006.

[13] E. J. Marinissen, V. Iyengar, and K. Chakrabarty. A Set of Benchmarks for Modular Testing of SOCs. In *Proc. ITC*, pp. 519–528, 2002.

[14] H. Murata, et al. Rectangle-Packing-Based Module Placement. In *Proc. ICCAD*, pp. 472–479, 1995.

[15] M. Nourani, M. Tehranipoor, and N. Ahmed. Pattern Generation and Estimation for Power Supply Noise Analysis. In *Proc. VTS*, pp. 439–444, 2005.

[16] H. F. Qian, S. R. Nassif, and S. Sapatnekar. Power Grid Analysis using Random Walks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24:1204–1224, August 2005.

[17] S. Ravi. Power-aware test: Challenges and Solutions. In *Proc. ITC*, pp. 1–10, 2007.

[18] S. Remersaro, et al. Scan-Based Tests with Low Switching Activity. *IEEE Transactions on Computers*, 24(3):268–275, June 2007.

[19] R. A. Rohrer, L. T. Pillage, and C. Wisweswariah. *Electronic and System Simulation Methods*. McGraw-Hill Book Company, 1995.

[20] J. Saxena, et al. A Case Study of IR-Drop in Structured At-Speed Testing. In *Proc. ITC*, pp. 1098–1104, 2003.

[21] C. Tirumurti, et al. A Modeling Approach for Addressing Power Supply Switching Noise Related Failures of Integrated Circuits. In *Proc. DATE*, pp. 1078–1083, 2004.

[22] P. Varma and S. Bhatia. A Structured Test Re-Use Methodology for Core-Based System Chips. In *Proc. ITC*, pp. 294–302, 1998.

[23] J. Wang and D. M. Walker. Modeling Power Supply Noise in Delay Testing. *IEEE Transactions on Computers*, 24(3):226–233, June 2007.

[24] J. Wang, et al. A Vector-based Approach for Power Supply Noise Analysis in Test Compaction. In *Proc. DATE*, paper 22.2, 2005.

[25] K. Wang and M. Sadowska. On-chip power-supply network optimization using multigrid-based technique. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24:407–417, Martch 2005.

[26] X. Wen, et al. Critical-Path-Aware X-Filling for Effective IR-Drop Reduction in At-Speed Scan Testing. In *Proc. DAC*, pp. 527–532, 2007.

[27] Q. Xu, N. Nicolici and K. Chakrabarty. Test Wrapper Design and Optimization under Power Constraints for Embedded Cores with Multiple Clock Domain. *IIEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26:1539–1547, August 2007.

[28] Q. Xu and N. Nicolici. Resource-Constrained System-on-a-Chip Test: A Survey. *IEE Proceedings, Computers and Digital Techniques*, 152(1):67–81, January 2005.

[29] M. Zhao, R. Panda, S. Sapatnekar, and D. Blaauw. Hierarchical Analysis of Power Distribution Networks. *IEEE Transactions on Computer-Aided Design*, 21(2):159–168, February 2006.