# A Scalable Soft Spot Analysis Methodology for Compound Noise Effects in Nano-meter Circuits

Chong Zhao
Department of ECE
Univ. of California, San Diego
La Jolla, CA 92093
(858) 534-7883
chong@ece.ucsd.edu

Xiaoliang Bai
Department of ECE
Univ. of California, San Diego
La Jolla, CA 92093
(858) 534-7883
xibai@ece.ucsd.edu

Sujit Dey
Department of ECE
Univ. of California, San Diego
La Jolla, CA 92093
(858) 534-0750
dey@ece.ucsd.edu

## ABSTRACT

Circuits using nano-meter technologies are becoming increasingly vulnerable to signal interference from multiple noise sources as well as radiation-induced soft errors. One way to ensure reliable functioning of chips is to be able to analyze and identify the spots in the circuit which are susceptible to such effects (called “soft spots” in this paper), and to make sure such soft spots are “hardened” so as to resist multiple noise effects and soft errors. In this paper, we present a scalable soft spot analysis methodology to study the vulnerability of digital ICs exposed to nano-meter noise and transient soft errors. First, we define “*softness*” as an important characteristic to gauge system vulnerability. Then several key factors affecting *softness* are examined. Finally an efficient Automatic Soft Spot Analyzer (ASSA) is developed to obtain the *softness* distribution which reflects the unbalanced noise-tolerant capability of different regions in a design. The proposed methodology provides guidelines to reduction of severe nano-meter noise effects caused by aggressive design in the pre-manufacturing phase, and guidelines to selective insertion of on-line protection schemes to achieve higher robustness. The quality of the proposed soft-spot analysis technique is validated by HSPICE simulation, and its scalability is demonstrated on a commercial embedded processor.

## Categories and Subject Descriptors

B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-tolerance.

**General Terms:** Reliability, Measurement, Design, Verification.

**Keywords:** Nano-meter technology, compound noise effect, robustness, softness distribution

## 1. INTRODUCTION

Reliable functioning of chips using nano-meter technologies is being severely challenged. Due to the shrinking of feature size and significant reduction in noise margins, nano-meter circuits are becoming more vulnerable to noise effects like crosstalk, IR drop, ground bounce, as well as radiation-induced transient soft errors. *To make matters significantly worse, as will be shown in this paper, different noise sources, and on-line error sources like radiation, interact with each other to create compound effects.*

Significant research and technology development has taken place to ensure signal integrity of nano-meter chips. Analysis and optimization techniques for crosstalk [2], IR-drop [3], ground bounce [4] and substrate noise [6] in System-on-Chips (SoCs) have been widely investigated. In addition, impacts of process variations [5] have drawn increasing research interest because large uncertainties in device parameters can lead to yield and performance degradation. However, most of the noise analysis and prevention techniques apply to a single source of noise, and cannot take into account the evolving reality of multiple noise sources interacting with each other. Moreover, even the single noise analysis techniques are very computational intensive and their applicability to complex system-on-chips is limited by their lack of scalability.

Nano-meter circuits are also becoming more vulnerable to radiation effects and other sources of soft errors. Single-event-upsets (SEUs) caused by cosmic ray neutrons or $\alpha$-particles [7] severely impact field-level product reliabilities [1]. Simulation-based methods have been adopted to analyze SEUs [8] but they are prohibitively time consuming. Error detection [9] and circuit hardening techniques [10] have been developed to protect circuits from SEUs. While providing feasible solutions, they require significant time/area overhead. Recently, the idea of partial protection was proposed [11] but efficient methodology of evaluating circuit robustness was not developed. Therefore, the selection of the spots to be hardened may not be optimal and the most vulnerable region in a design might not be protected, resulting in an insufficient partial protection solution.

Although extensive research has been conducted on individual noise sources, to the best of our knowledge, there have been few attempts to develop analysis techniques for combined noise/soft error effects. As we will demonstrate in section 2, different noise sources tend to aggravate the effects of each other, making stand-alone noise analysis results inaccurate. Furthermore, due to the unpredictable and transient nature of these noise effects, on-line detection and protection may become inevitable as 100% robustness can no longer be guaranteed solely by design. However, blindly applying hardening techniques to the entire design incurs unacceptable design overhead. Therefore, a methodology that evaluates compound noise effects and identifies the regions of vulnerabilities (soft spots) is becoming essential to guide efficient and economical robust digital system design.

This work was supported partially by the MARCO/DARPA Gigascale Systems Research Center (GSRC)

In this paper, we propose a soft spot analysis methodology targeting at compound noise effects. Fundamentally different from approaches focused on noise behavior, our approach focuses on analyzing the ***noise-immunity*** of the design. We evaluate the vulnerability of different portions of a circuit based on accurate timing, logic and electrical information available during the design phase. Using our methodology, we are able to provide an overall vulnerability distribution in a design and discover that the inherent noise-tolerant abilities of different nodes greatly vary. This distribution allows the designers to further investigate the most vulnerable nodes, to eliminate potential noise effects in the design phase, and to selectively apply on-line hardening techniques. As a result, the cost and effort of designing highly robust circuits can be dramatically reduced. Our methodology is a static approach that does not require dynamic simulation or intensive computation. Numerical results show that the proposed methodology is accurate, efficient and scalable to large complex designs.

The rest of this paper is organized as follows. In section 2, multiple noise source induced failures are studied. Section 3 describes the static soft spot analysis in detail. In section 4, experimental results are presented. Section 5 concludes this paper.

## 2. Multiple Noise Source Induced Failures

Multiple noise effects exist in a nano-meter circuit and they may combine with each other to severely affect circuit performances. **Figure 1** shows several different sources of noise in a nano-meter circuit including: crosstalk caused by a switching neighboring wire, IR-drop from resistive power grid, large AC current together with inductance-induced ground bounce, substrate coupling noise and particle-strike-induced soft errors. Each of these effects can individually cause circuit failures. As we will show through simulation, different noise sources can also combine to magnify their effects, greatly increasing the possibility of errors.

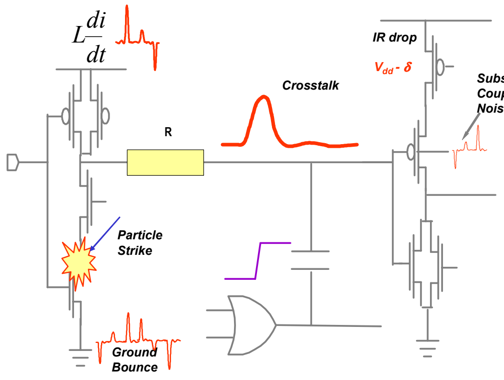


**Figure 1. Multiple Noise Sources in a Nano-meter Circuit**

In the circuit shown in **Figure 2**, due to coupling capacitance between the victim and each aggressor, when the input of INV1 remains at 0, a 0→1 transition at the input of INV3 or INV5 will result in a negative crosstalk glitch on the victim net. However, as shown by the INV2 input (xv) and output (dv) voltages in Figure 3(a), even in the worst case when both aggressors switch simultaneously in the same direction, the output glitch of INV2 is not strong enough to cause a logic error in DFFV. This is due to the crosstalk noise on the victim being attenuated by INV2, so the value in DFFV remains at "0" (the qv curve in **Figure 3**(a)). However, if a particle strikes on ***either*** INV1 (**Figure 3**(b)) or INV2 (**Figure 3**(c)) at a proper time, with all the other conditions unchanged, an erroneous "1" is latched in DFFV. In our experiment, soft errors are modeled as current sources [12] between the drain of the MOS transistor and the output node. In another case, a glitch that would not have been strong enough to change the state of DFFV is turned into a latched error by an IR-drop on the power line (Figure 3 (d)).

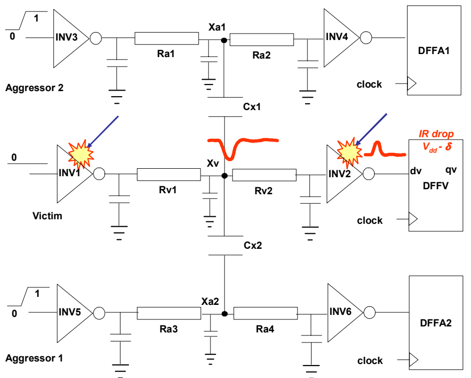


**Figure 2. Compound Noise Effects: Example Circuit**

These examples demonstrate that although the essential physical mechanisms of these noise effects are different, they can affect circuit behavior in a combined manner. Therefore, a system exposed to multiple noise sources becomes more vulnerable. As a result, a methodology trying to address effect of one single noise source while ignoring others may be overly optimistic. Next, we will propose a promising solution, soft spot analysis methodology, which considers multiple noise sources and is able to efficiently evaluate of the vulnerability of a given design.

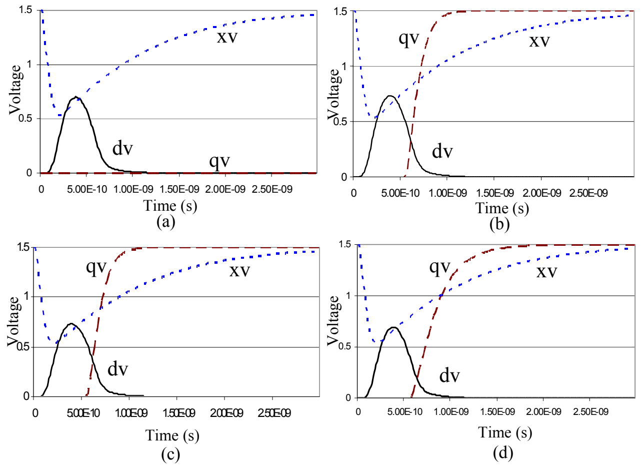


**Figure 3. Compound Noise Effects: HSPICE Simulation**

## 3. The Soft Spot Analysis Methodology

As shown in section 2, the random occurrences and complex physical mechanisms of noise depend on many factors that cannot be precisely determined until the product is manufactured and operating in a real environment. On the other hand, accurate information about the design, such as timing feature, logic paths and layout-extracted electrical characteristics, can be obtained by performing various analyses during the design phase. Therefore, if we fully utilize such available information to study the noise-immunity of the design, we will be able to locate soft spots that are vulnerable under the attack of unpredictable noise sources. Based on this fact, we propose a soft spot analysis methodology.

Given a design, we choose the output nodes of all logic gates to be the target of our investigation. Each "node" is used to represent the gate that drives it as well as the nets it drives. Then, **for each node $N$, we define "*softness*" $S_N$ as its vulnerability to noise, reflected by its tendency to allow noise to propagate through it with enough strength and proper timing to eventually cause observable errors.** An observable error refers to one that is latched into a memory element and thus can become a stable erroneous logic value. Our objective is to determine the magnitude of $S_N$ for all nodes. By identifying nodes with large values of $S_N$ as the most vulnerable ones in a design, further analysis and improvement can be carried out in a well studied and guided rather than blind manner.

Not all noise can eventually cause errors due to three masking effects: timing masking, electrical masking and logic masking. Each masking effect tends to prevent a noise from causing observable error effects. Therefore, $S_N$ must correspondingly reflect all three masking effects at node $N$. Next, we introduce novel methods to obtain numerical representations of all three masking effects, and then discuss how to calculate $S_N$ based on the strengths of these masking effects.

## 3.1. Timing Masking

Timing masking means that a noise can cause an observable error only if it reaches a memory element within its sampling window. For a DFF, the sampling window is bounded by the setup time ($t_{su}$) and hold time ($t_h$) around the active clock edge, as shown on the right-hand side of Figure 4. In order to decide the required time interval for noise at a node to reach any DFF within its sampling window, we define the *effective noise window* $TW_{eff}^N$ as such that only noise existing at node $N$ inside $TW_{eff}^N$ may be able to reach at least one DFF during the DFF's sampling window. In other words, if a noise originates or arrives at node $N$ before the start (after the end) of $TW_{eff}^N$, it will reach all DFFs before the start (after the end) of their sampling window, and will therefore not be captured by any DFF. As shown in **Figure 4**, $TW_{eff}^N$ of a specific path $p$ is bounded by a start time $t_{start}^{Np}$ and an end time $t_{end}^{Np}$, decided by the worst-case longest delay $(\Delta T^p)_{max}$ and best-case shortest delay $(\Delta T^p)_{min}$ from node $N$ to the DFF through path $p$, respectively. If the clock period is $T$, we have:

$$t_{start}^{Np} = T - t_{su} - (\Delta T^p)_{\max}$$
$$t_{end}^{Np} = T + t_h - (\Delta T^p)_{\min}$$

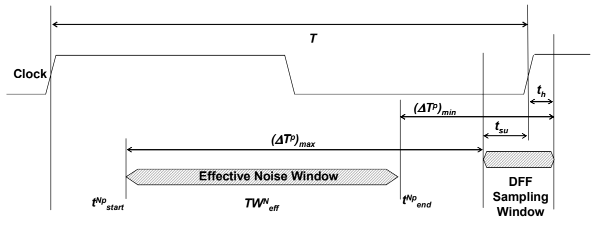


**Figure 4. Calculating Effective Noise Window**

We then use the maximum (latest) $t_{end}^{Np}$ and the minimum (earliest) $t_{start}^{Np}$ among all paths to calculate a pessimistic $TW_{eff}^N$ of node $N$ to all DFFs that node $N$ is logically connected to. Let P be the collection of all possible paths through node N:

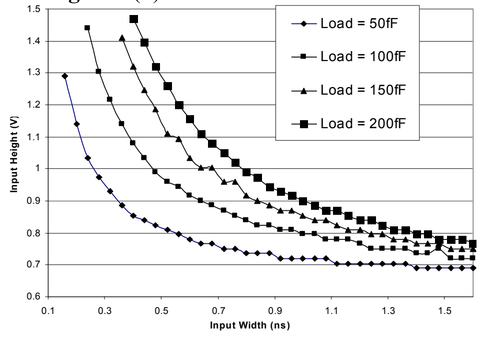
$$TW_{eff}{}^N = \max_{p\in P}\{t_{end}^{Np}\} - \min_{p\in P}\{t_{start}^{Np}\} \qquad (1)$$

A larger $TW_{eff}^N$ means a higher probability that a noise on node $N$ can reach a DFF within the DFF's sampling window. Hence, we can use $TW_{eff}^N$ as a measurement of the timing masking effect at node $N$.

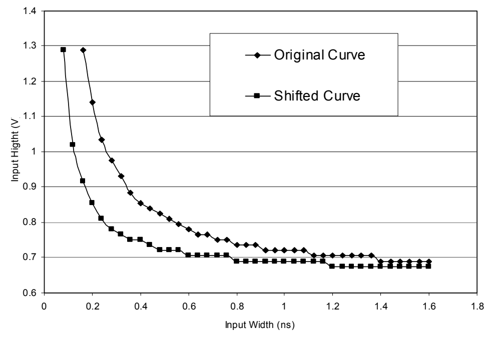
## 3.2. Electrical Masking

Electrical masking means that a noise must have enough duration and amplitude to propagate through a logic gate without being completely attenuated. The strength of the electrical masking effect can be characterized by a *noise rejection curve*. **Figure 5**(a) shows an example of noise rejection curves of an inverter in a 0.18μm cell library with different capacitive loads. The X and Y axes are the width and height of the input noise, respectively, while a glitch can propagate through the gate only if its shape is in the region above the curve.

Noise rejection curves can be viewed as a representation of the ability of a gate to filter noise caused by any source. In reality, the natures of some kind of noise may be completely unknown during design, while the effects of some other kinds of noise, such as crosstalk, can be conveniently estimated. If we can build information of the "known" noise into the noise rejection curve, we will obtain curves that reflect the "remaining" noise margin to the "unknown" noise. We present an idea of "curve shift" to realize this idea. As an example, a gate whose input is closely coupled with other nets may experience certain crosstalk, so its tolerance to other noise is reduced, which means a glitch propagating from somewhere else may not be electrically masked even its shape is below the pre-calibrated curve. In another point of view, the noise rejection curve "shift" toward the axes. As an example, a curve shift caused by crosstalk of the same inverter is shown in **Figure 5**(b).



(a) Noise Rejection Curve of an Inverter



(b) Curve Shift Due to Crosstalk

**Figure 5. Noise Rejection Curve and Curve Shift**

Next, we utilize the noise rejection curve and the concept of curve shift to measure the electrical masking effect. Observing that a smaller area under the curve indicates noise with smaller height and width can propagate through, we use the ratio of the area above the curve to the area below the curve, denoted as $R_e^N$, as a measurement of electrical masking effect at node $N$. The total area of the noise rejection curve is determined by the product of the power supply voltage and the maximum input glitch width that can be assumed to be the clock period used in the design. A larger $R_e^N$ means a larger portion of the input noise can propagate through the gate without being electrically masked.

To calculate $R_e^N$, we first calibrate the standard cell library using HSPICE to create a database of noise rejection curves for all gates with different capacitive loads. Then for a given design, we calculate the load capacitance of each node to retrieve the proper noise rejection curve from the database. Crosstalk effects are then estimated using the extended 2-π model [13] and are used to shift the retrieved noise rejection curve. Finally $R_e^N$ is obtained by computing the area ratio for the modified curve. Curve shifts caused by other mechanisms can be similarly considered, given specific design and process information.

By using curve shifting, we avoid calculating the exact noise caused by multiple sources, but we are still able to consider as much available information as possible in a single quantity $R_e^N$. An obvious advantage is that cell calibration is done only once to obtain noise rejection curves for a given library and can be used on all circuits implemented in the same library. As a result, we can fully make use of the accuracy of HSPICE without repeatedly suffering from its time-consuming nature.

## 3.3. Logic Masking

Logic masking refers to the effect that noise will cease to propagate through a gate whose output is solely determined by its other inputs. Complete evaluation of the logic masking effect requires exhaustive exploration of the entire input vector space, which is not feasible for complex designs. As an alternative, we developed an efficient logic path tracing technique to calculate **the probability of a noise propagating from node *N* to DFFs through legitimate logic paths**, denoted by $P_{prop}^N$. Our algorithm consists of two steps; each using a breadth-first approach to go through the design netlist.

The first step of our technique is to derive the probability for node $N$ being logic "1", denoted by $Pr^N(1)$, ($Pr^N(0)$ is simply $1-Pr^N(1)$). The example shown in **Figure 6** best illustrates how to calculate $Pr^N(1)$ for a circuit. We assume the probabilities for all inputs ($Pr^A(1)$, $Pr^B(1)$, $Pr^C(1)$, and $Pr^D(1)$) are 1/2. Knowing that for a NAND gate, the output is "1" if any of the inputs is "0"; for a NOR gate, the output is "1" only if all inputs are "0"s; and for an INV, the output is "1" if the input is "0", the resulting $Pr^E(1)$, $Pr^F(1)$, $Pr^G(1)$, and $Pr^H(1)$ are listed in Figure 6 above the circuit. For a given design, if we know the probability of logic 1 (or 0) at each primary input (PI) $i$, $P_{pi}^i(1)$ (or $P_{pi}^i(0)$), we can systematically trace forward from all PIs throughout the entire netlist to calculate $Pr^N(1)$ for each node $N$. The probability of different logic values at PIs can be obtained through functional vector simulations. If no such information is available, a good approximation is to assume $P_{pi}^i(1)= P_{pi}^i(0)=$ 1/2 for every PI.

In the second step, in order to derive the propagation probability for node $N$, we trace back along all logic paths from DFFs. For every gate $M$ encountered during the trace-back of path $p_j$ from DFF $j$, we calculate $P_M^{NC}$, the probability that **all inputs of gate *M* that are NOT on path $p_j$ carry non-controlling values**, (For example, the non-controlling value is 1 for a NAND gate and 0 for a NOR gate), using results from the first step. $P_M^{NC}$ thus equals to the probability for a noise on path $p_j$ **NOT** to be logically masked by gate $M$. When node $N$ is reached, the product of all $P_M^{NC}$ gives $P_{prop}^N(p_j)$, the probability for noise at node $N$ to propagate to DFF $j$ through path $p_j$. In **Figure 6**, we show some of the calculated propagation possibilities below the circuit.

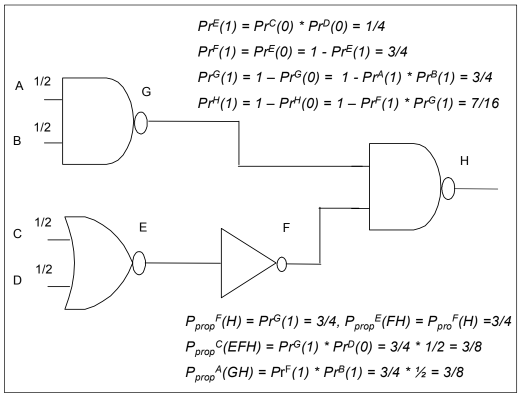

$Pr^E(1) = Pr^C(0) * Pr^D(0) = 1/4$
$Pr^F(1) = Pr^E(0) = 1 - Pr^E(1) = 3/4$
$Pr^G(1) = 1 – Pr^G(0) = 1 - Pr^A(1) * Pr^B(1) = 3/4$
$Pr^H(1) = 1 – Pr^H(0) = 1 – Pr^F(1) * Pr^G(1) = 7/16$
A
1/2
B
1/2
G
H
C
1/2
D
1/2
E
F
$P_{prop}^F(H) = Pr^G(1) = 3/4$, $P_{prop}^E(FH) = P_{pro}^F(H) =3/4$
$P_{prop}^C(EFH) = Pr^G(1) * Pr^D(0) = 3/4 * 1/2 = 3/8$
$P_{prop}^A(GH) = Pr^F(1) * Pr^B(1) = 3/4 * ½ = 3/8$

**Figure 6. Example of Static Logic Path Tracing Technique**

Finally, we sum $P_{prop}^N(p_j)$ over all possible paths $p_j$ to obtain the propagation probability $P_{prop}^N$. Furthermore, in reality, all DFFs in a design may not be of equal functional significance. This can be taken into account by assigning DFF $j$ a weighting factor $F_j$, $P_{prop}^N$ can thus be expressed as:

$$P_{prop}{}^N = \sum_{j=1}^{M} F_j * \sum_{p_j} P_{prop}{}^N(p_j) \qquad (2)$$

where the second summation is over all paths from node $N$ to DFF $j$ and $M$ is the total number of DFFs. The weighting factors can be provided by designers based on application-specific knowledge. For example, a DFF that decides key state transitions in a state machine design may be assigned a large weighting factor.

We use $P_{prop}^N$ to measure the logic masking effect at node $N$: a larger $P_{prop}^N$ indicates a higher chance for noise at node $N$ to logically reach a DFF. Our logic path tracing technique provides more comprehensive coverage than methods that only relies on limited functional simulations and methods that simply assign larger weights to nodes close to DFFs. Its accuracy and flexibility are further improved by the introduction of input logic value probabilities and functional weighting factors.

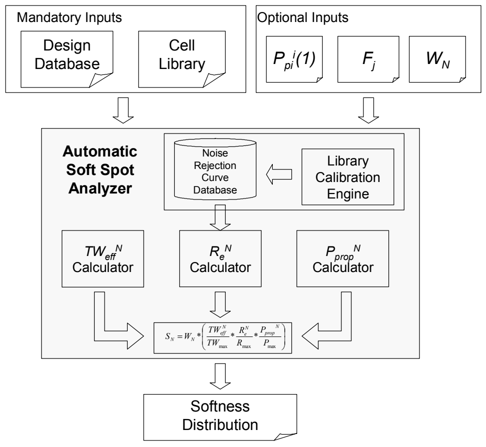
## 3.4. Evaluating *Softness*

As we have discussed, *softness* $S_N$ is a measure of the vulnerability of node $N$ when exposed to compound noise effects, and should be a function of $TW_{eff}^N$, $R_e^N$ and $P_{prop}^N$. While $S_N$ may have many possible forms, if $TW_{eff}^N$, $R_e^N$ and $P_{prop}^N$ are considered to contribute independently, $S_N$ can be expressed as:

$$S_N = W_N * \left( \frac{TW_{eff}^N}{TW_{\max}} * \frac{R_e^N}{R_{\max}} * \frac{P_{prop}{}^N}{P_{\max}} \right) \qquad (3)$$

where $TW_{max}$, $R_{max}$, and $P_{max}$ are the maximum values among all nodes used as normalization factors. $W_N$ is another application-specific weighting factor, which provides another dimension of flexibility by allowing designers to denote certain portions of the design as more critical than others.

In order to determine the numerical values of $S_N$, we have developed an automated flow called "Automatic Soft Spot Analyzer" (ASSA) shown in **Figure 7**. To execute, ASSA first uses a "Library Calibration Engine" to generate a noise rejection curve database for the cell library. Note that this step only needs to be performed once for each library, after which the database may be read in from storage when analyzing a given design. Next, timing, electrical and logic masking factors are evaluated using information available in the design database (gate-level netlist, timing, physical layout, extracted RC, *etc.*). The designer may also provide application-specific information as optional inputs including input logic probabilities, DFF functional weighting factors and overall softness weighting factors. ASSA then calculates $S_N$ and provides a *softness* distribution as its output.



Mandatory Inputs
Design Database
Cell Library
Optional Inputs
$P_{pi}^{i}(1)$
$F_j$
$W_N$
Automatic Soft Spot Analyzer
Noise Rejection Curve Database
Library Calibration Engine
$TW_{eff}^{N}$ Calculator
$R_e^{N}$ Calculator
$P_{prop}^{N}$ Calculator
$S_N = W_N * \left( \frac{TW_{eff}^{N}}{TW_{\max}} * \frac{R_e^N}{R_{\max}} * \frac{P_{prop}^{N}}{P_{\max}} \right)$
Softness Distribution

**Figure 7 Automatic Soft Spot Analyzer (ASSA)**

Next we will demonstrate the efficiency, accuracy and scalability of ASSA through extensive experiments.

## 4. Experimental Results

We applied the ASSA methodology to four sample circuits to evaluate the quality and speed by comparing the results with HSPICE simulation. Two of the four circuits are basic blocks in many digital designs (ADDER: 4-bit adder and DEC: 4-bit decoder), the other two (Xt1 and Xt2) are random logics extracted from a commercial processor (Xtensa$^{TM}$ from Tensilica [14]). All circuits are combinational blocks with registered primary outputs and are synthesized by Synopsys DesignCompiler$^{TM}$ using a 0.18μm cell library; Cadence SiliconEnsemble$^{TM}$ is used for generating physical layout; Mentor Graphics Xcalibre$^{TM}$ is used to extract RC networks; and Synopsys PrimeTime$^{TM}$ is used for static timing analysis. In these experiments, we explicitly considered glitch-type noise caused by crosstalk and particle strike. However, the proposed methodology is also applicable to other noise effects.

In these experiments, our goal is to measure the vulnerability of the internal nodes through accurate SPICE simulation, and to compare the result with ASSA calculation. The approach is to focus on a specific node at a time – run a large number of input vectors while spuriously injecting soft errors caused by particle strikes (using the current source model [12]) on that node. These soft errors interact with crosstalk noise on the node due to large coupling capacitances extracted from layout to produce compound effects. The number of observable errors caused by noise on this node is counted. At the end of simulation the error count will be used as the “simulated softness” of the node.

These results were compared with the softness values computed using ASSA. During the computation, the input logic probability $P_{pi}^{i}(1)$ reflects the actual statistics of HSPICE simulation input vectors; the other two optional inputs ($F_j$ and $W_N$) were set to 1 to produce application-independent softness distributions.

**Table 1** shows some statistics about the experiments. Row 1 is the area and row 2 is the internal node count of the sample circuits. Row 3 and row 4 show the number of nodes we chose to inject noise on and the number of input vectors used for simulating effects of noise on each node. Row 5 compares the runtime of ASSA to that of HSPICE where the numbers shown are the average evaluation time for each node: speedup factors (shown in row 6) at the order of $10^3$ were achieved by ASSA over HSPICE. Furthermore, as the circuit complexity increases, HSPICE simulation speed decreased drastically so tradeoffs had to be made between precision and runtime. For example, the size of DEC is only 2.3 times of the size of ADDER, but the number of chosen nodes and simulated vectors had to be reduced to 1/2 and 1/4, respectively, in order to finish simulation in comparable time. On the contrary, ASSA shows constant analyzing time for circuits of similar complexity (ADDER, Xt1 and Xt2) while exhibits performance improve as the circuit size increases (DEC). This improvement is caused by the constant tool initialization time that is independent of the circuit size.

**Table 1: Sample Circuits and Simulation Time**

| | | ADDER | DEC | Xt1 | Xt2 |
|---|---|---|---|---|---|
| 1. Area (Library Unit) | | 1107 | 2488 | 865 | 995 |
| 2. Node Count | | 89 | 210 | 74 | 59 |
| 3. No. Simulated Nodes | | 42 | 21 | 23 | 30 |
| 4. No. Input Vectors | | 512 | 128 | 512 | 256 |
| 5. Runtime (sec/node) | HSPICE | 12,062 | 19,097 | 9,548 | 5,230 |
| | ASSA | 4.65 | 2.45 | 4.14 | 4.88 |
| 6. Speedup Factor | | 2.6x$10^3$ | 7.8x$10^3$ | 2.3x$10^3$ | 1.1x$10^3$ |

The calculated softness was compared with the simulated softness node by node. **Figure 8** shows the results: the x-axis is the indices of the simulated nodes; the y-axis is the normalized softness, where the “ASSA” and “HSPICE” curves show the calculated and the simulated values, respectively. It can be seen that ASSA not only correctly captured the most vulnerable nodes but also provided a distribution of softness among all simulated nodes that matched HSPICE results very well.

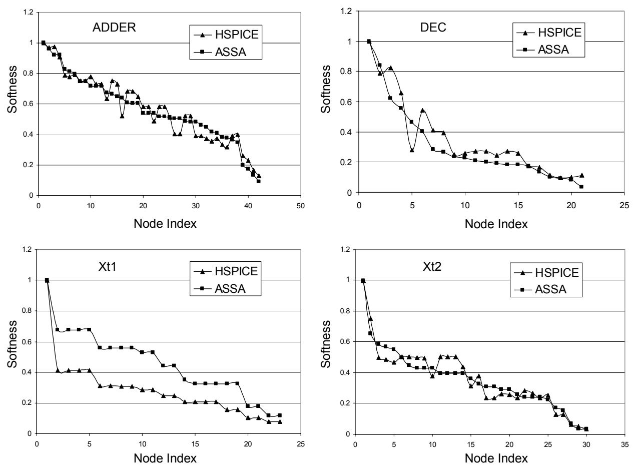


**Figure 8. ASSA Results Compared with HSPICE Simulation**

It is worth mentioning that the goal of ASSA is not trying to find the most vulnerable one or two nodes, or to precisely compare the vulnerability of two specific nodes. Instead, it is to identify a collection of most vulnerable nodes. Therefore, although individual discrepancies do exist, it does not degrade the quality and significance of the proposed methodology. Even for the circuit Xt1, where an apparent gap between the simulated and calculated values, high degree of correlation exists in the softness distribution in the sense that ASSA correctly captures the relative vulnerability among the nodes.

We then applied ASSA to the Xtensa$^{TM}$ processor, a commercial state-of-the-art configurable and extensible RISC processor. We conducted experiments on a large logic module

EX, which has 97 registered output ports, 338 input ports and 3156 internal nodes. EX is particularly challenging because of its large number of inputs, unbalanced logic paths and strong crosstalk effects due to aggressive layout scheme. Due to the performance and reliability requirements, it is essential to identify and fix as many potential vulnerable spots as possible and provide low-cost on-line protection schemes in early design phase. Simulation-based methods are not applicable because of the design complexity. Our soft spot analysis, on the other hand, promptly pinpointed the most vulnerable nodes. **Table 2** shows a breakdown of time taken in each step. Note that the processing time of each node is comparable to the design DEC, indicating that our methodology is able to scale approximately linearly as the design complexity increases.

**Table 2: Soft Spot Analysis Performance**

| Operation | Time Taken |
|---|---|
| Calculating $R_e^N$ | 52min* |
| Calculating $P_{prop}^N$ | 2min |
| Calculating $TW_{eff}^N$ | 78min |
| Calculating *Softness* $S_N$ | 2min |
| **Total Time** | 134min |
| **Process Time Per Node** | 2.55 sec |

**: Library calibration time not included.*

**Figure 9** shows the *softness* distribution of all nodes in design EX: the x-axis is the normalized *softness* and the y-axis is the number of nodes with different *softness* values. It clearly demonstrates the non-uniform *softness* distribution among all nodes: only about 0.7% (22) has *softness* larger than 10% of the maximum value, and another 18.6% (587) has *softness* larger than 1%, whereas the *softness* values of the other 80.7% of all nodes are at least 2 orders of magnitude lower than the maximum value.

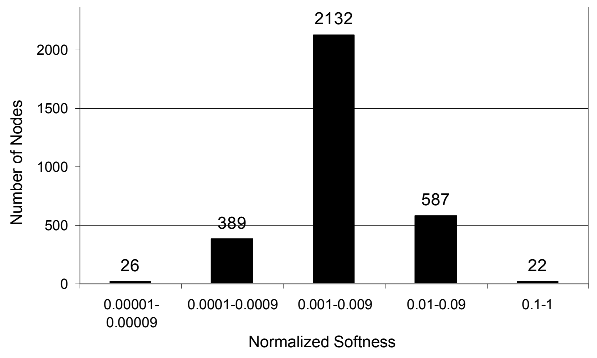


**Figure 9. Softness Distribution in Design EX**

The proposed soft spot analysis methodology has extensive applications. The unbalanced softness distribution is crucial in efficient, low-cost robust circuit design. During the pre-manufacturing design phase, accurate but time consuming analysis can be subsequently performed on the identified most vulnerable spots. If the potential severe noise effects are caused by aggressive design, design modifications can be done to remedy the *softness* of that portion of the circuit. Furthermore, our methodology provides a guideline to selectively apply on-line error detection and protection scheme to the spots that are most likely affected by transient errors during its lifetime, so a high degree of on-line robustness can be achieved with low design overhead.

## 5. Conclusion and Future Work

In this paper, an efficient soft spot analysis methodology is proposed to identify the most vulnerable spots in a design exposed to multiple noise sources. It has been validated with extensive HSPICE simulations. We also applied it to a major block in a commercial configurable processor to demonstrate its efficiency and scalability in analyzing complex designs.

An important advantage of the proposed methodology is that it is an open framework: with reasonable efforts, effects of more noise sources can be taken into consideration using similar approaches.

The proposed methodology will greatly facilitate robust system design by guiding both design and on-line optimization. In order to demonstrate its application, one of our current projects involves design improvement and robustness insertion under the guidance of soft spot analysis.

## References

[1] Semiconductor Industry Association, International Technology Roadmap for Semiconductors, 2001.

[2] J. Cong, D. Z. Pan, and P. V. Srinivas, "Improved crosstalk modeling for noise constrained interconnect optimization," *Proc. ASP-DAC*, *pp*. 373-378, 2001.

[3] Young S. "Identifying IR drop in high performance nanometer design," *Electronic Engineering, vol.74, no.905*, *pp*. 30-33, June 2002.

[4] Shen Lin, Chang N., "Challenges in power-ground integrity," *IEEE Intl. Conf. Computer Aided Design*, *pp*. 651-644, 2001.

[5] Hess C, Stine BE, Weiland LH, Sawada K., "Logic characterization vehicle to determine process variation impact on yield and performance of digital circuits," *Intl. Conf. Microelectronic Test Structures, pp*. 189-196, 2002.

[6] Hongmei Li, Carballido J, Yu HH, Okhmatovski VI, Rosenbaum E, Cangellaris AC., "Comprehensive frequency-dependent substrate noise analysis using boundary element methods," *Intl. Conf. Computer Aided Design*, *pp*. 2-9, 2002.

[7] Peter Hazucha, Christer Svensson, "Cosmic-Ray Soft Error Rate Characterization of a Standard 0.6-µm CMOS Process," IEEE *Jnl. Solid-State Circuits*, vol. 35, no. 10, Oct. 2000.

[8] IROC Technologies, http://www.iroctech.com

[9] Anghel, L., Nicolaidis, M., "Cost Reduction and Evaluation of a Temporary Faults Detecting Technique," *DATE'00*, *pp*. 591-598, March 2000.

[10] Y. Zhao and S. Dey, "Separate-Dual-Transistor (SDT) Register-An on-line Testing Solution for Soft Errors in UDMS-IC," *IOLTS*, Kos Island, Greece, 2003.

[11] K. Mohanram, N.A. Touba, "Cost-Effective Approach for Reducing Soft Error Failure Rate in Logic Circuits," *IEEE International Test Conference*, *pp*. 893-901, Sept., 2003.

[12] L. B. Freeman, "Critical charge calculations for a bipolar SRAM array," *IBM J. Res. Dev., Vol. 40, pp. 119-129*, Jan, 1996.

[13] X. Bai, R. Chandra, S. Dey and P.V. Srinivas, "Noise-Aware Driver Modeling for Nanometer Technology," *IEEE International Symposium on Quality Electronic Design,* ISQED'03, March 2003, pp.177~182.

[14] http://www.tensilica.com/xtensa_overview_handbook.pdf, *Xtensa$^{TM}$ Microprocessor Overview Handbook*, Tensilica Inc, August 2001.