Harmony: Static Noise Analysis of Deep Submicron Digital Integrated Circuits

Kenneth L. Shepard, *Member, IEEE*, Vinod Narayanan, *Senior Member, IEEE*, and Ron Rose

***Abstract*—As technology scales into the deep submicron regime, noise immunity is becoming a metric of comparable importance to area, timing, and power for the analysis and design of very large scale integrated (VLSI) systems. A metric for noise immunity is defined, and a static noise analysis methodology based on this noise-stability metric is introduced to demonstrate how noise can be analyzed systematically on a full-chip basis using simulation-based transistor-level analysis. We then describe Harmony, a two-level (macro and global) hierarchical implementation of static noise analysis. At the macro level, simplified interconnect models and timing assumptions guide efficient analysis. The global level involves a careful combination of static noise analysis, static timing analysis, and detailed interconnect macromodels based on reduced-order modeling techniques. We describe how the interconnect macromodels are practically employed to perform coupling analysis and how timing constraints can be used to limit pessimism in the analysis.**

***Index Terms*—Deep-submicron IC's, interconnect coupling, signal integrity, static noise analysis.**

I. Introduction

NOISE has traditionally been a concern to analog designers, since it represents a lower bound on the magnitude of a signal that can be usefully amplified. It also presents an upper bound to the useful gain of an amplifier, since noise will ultimately saturate an amplifier if the gain is too high. The noise sources of concern in analog design derive from physical sources—thermal noise, flicker noise, and shot noise, for example. These *physical noise sources* come about because of the discreteness of electronic charge and the stochastic nature of electronic transport processes [1].

In contrast, digital circuits, by virtue of the large, abrupt voltage swings characteristic of their operation, create deterministic man-made noise several orders of magnitude greater than noise from stochastic physical sources. Problems due to these noise sources were first observed in mixed-signal applications [2], [3], which plunged highly noise-sensitive analog circuits into a noisy digital environment. Although digital circuits create much more noise than analog circuits, digital systems are prevalent because they are inherently immune to noise. Until recently, noise immunity overcame the noisiness of digital circuits. Technology scaling and performance demands have unfortunately changed this balance, and noise is now a problem even in purely digital designs.

Noise has become a metric in the design of digital integrated circuits of comparable importance to area, timing, and power for four principle reasons: increasing interconnect densities, faster clock rates, more aggressive use of high-performance circuit families, and scaling threshold voltages. All of these factors degrade the signal-to-noise ratio for CMOS digital designs. Increasing interconnect densities imply a significant increase in coupling capacitance as a fraction of self-capacitance. Faster clock rates imply faster on-chip slew times. These two effects combine to make capacitive coupling a growing source of noise on-chip. Many high-performance circuit styles try to speed up one transition (usually falling) at the expense of the other and assign logical evaluates to the faster edge. Any circuit that utilizes these techniques we refer to as a *skewed-evaluate* circuit. Skewed-evaluate circuits have noise sensitivities directly related to the threshold voltages of the transistors responsible for the evaluate transitions [usually n-channel field-effect transistors (n-FET's)]. Threshold voltages are, however, scaling lower to maintain drive in the presence of scaling supply voltages. These effects combine to produce more sources of on-chip noise due to switching circuits as well as less immunity to this noise. More details of these technology trends can be found in [4].

Noise has two deleterious effects on digital design. When noise acts against a normally static signal, it can transiently destroy the logical information carried by the static node in the circuit. If this ultimately results in incorrect machine state stored in a latch, functional failure will result. When noise acts simultaneously with a *switching* node, this is manifest as a change in the timing (delay and slew) of the transition (a noise-on-delay effect). We are concerned with the former effect in this paper.

We present the first comprehensive methodology for understanding and analyzing the noise immunity of digital integrated circuits. There are three essential components of this static noise analysis: calculating noise due to coupling in the interconnects, calculating noise injected or propagated by the circuits, and having a criterion for deciding when the noise occurring on a node due to circuit and interconnect noise exceeds the noise immunity of the receiving circuits. In Section II, we introduce a noise classification based on the noise level relative to the supply and ground rails. We also describe the noise sources that are affecting digital design

Manuscript received August 29, 1997; revised July 3, 1998. The work performed at Columbia University was supported in part by the National Science Foundation (NSF) under Grant CCR-97-34216. This paper was recommended by Associate Editor K. Mayaram.

K. L. Shepard is with the Columbia Integrated Systems Laboratory, Department of Electrical Engineering, Columbia University, New York, NY 10027 USA.

V. Narayanan is with CadMOS Design Technology, San Jose, CA 95113 USA.

R. Rose is with IBM Microelectronics, Essex Junction, VT 05452 USA.

Publisher Item Identifier S 0278-0070(99)05683-3.

0278–0070/99$10.00 © 1999 IEEE

and that have to be included in the analysis. Section III describes the noise stability metric as a practical, formal basis for ensuring noise immunity. We describe the static noise analysis approach in Section IV, a technique for identifying all possible on-chip functional failures without full pattern-dependent dynamical simulation. In Section V, we describe Harmony, a two-level hierarchical implementation of static noise analysis, which combines static timing analysis [5] and reduced-order modeling with transistor-level analysis. We discuss the additional, specific assumptions used to guide an implementation used on a real design. Section VI provides a comprehensive example of static noise analysis and provides performance and memory usage statistics for Harmony as applied on an S/390 microprocessor design [6].

II. Noise Fundamentals

We define an *evaluation node* in a CMOS digital integrated circuit as any node that is used to carry information between the logic gates of the circuit. As such, these are usually the inputs and outputs of the channel-connected components (CCC's) of the design; that is, transistors connected through their sources and drains. Noise, then, is any deviation from the nominal supply or ground voltages at evaluation nodes which should otherwise represent stable logic one or zero. In digital circuits, analog voltages carry logical information used in computation. Although noise causes these analog voltages to vary, the system still functions as long as the voltages fall into a valid range. If this is not the case, then the circuits' correct functioning cannot be certain. The complexity of noise analysis comes from the fact that the voltage ranges that represent valid logic levels depend on the precise time-domain characteristics of the noise appearing on the evaluation nodes as well as the sensitivity of receiving circuits to this noise.

It is convenient to classify noise according to the voltages' relationship to the rails.

- V_H noise reduces an evaluation node voltage below the supply level.
- V_H^* noise increases an evaluation node voltage above the supply level.
- V_L noise increases an evaluation node voltage above the ground level.
- V_L^* noise decreases an evaluation node voltage below the ground level.

The supply and ground reference levels are presumed to be set from the external reference to the chip. We refer to V_H^* and V_L^* noise as *bootstrap* noise. These noise classifications (shown in Fig. 1) are useful because circuits generally propagate noise types in well-defined way. For example, a CMOS inverter is sensitive to V_L and V_H noise on its input, propagating it as V_H and V_L noise, respectively, to its output.

To develop a comprehensive strategy for noise analysis, we must consider all the possible sources of noise on-chip. Each of these sources is fundamentally due to the use of large-signal voltage changes to switch logic levels. These switching events interfere with static signals as shown in Fig. 2 because of coupling through the interconnect (coupling noise), through the transistors (charge-sharing noise or coupling noise across feedback device capacitances), through the substrate or n-well (substrate noise), or through the power supply (power-supply noise). In the remainder of this section, we briefly consider these noise sources to provide a basis for how they are handled in the context of static noise analysis. More details can be found in [4].

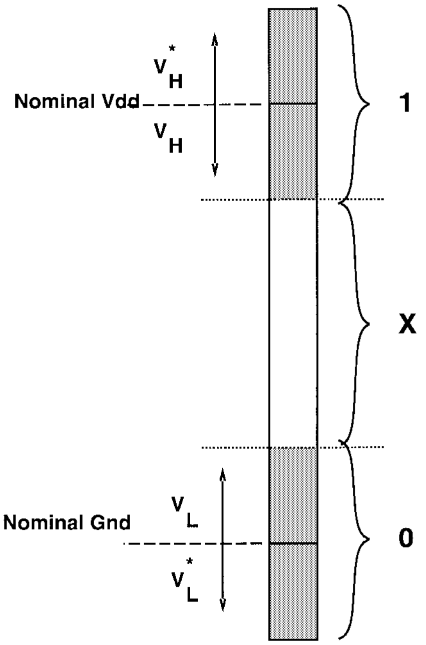


Fig. 1. A range of analog voltages defines the digital zero and one.

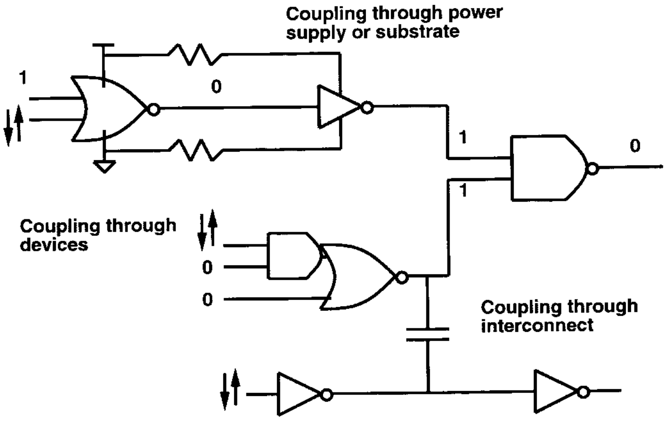

Coupling through power supply or substrate
1
0
1
0
1
Coupling through devices
0
0
Coupling through interconnect

Fig. 2. Switching events interfere with static signals because of coupling through the interconnect, through the transistors, through the substrate or n-well, or through the power supply.

A. Interconnect Coupling Noise

Coupling noise, or cross talk, is primarily due to capacitive coupling between metal lines [7]–[9]. Fig. 3 shows a highly simplified analysis (neglecting interconnect resistance) of the essential attributes of this noise. In Fig. 3(a), coupled noise on the victim evaluation node between the two inverters results from switching on the neighboring perpetrator line denoted by the voltage source. In the circuit representation in Fig. 3(b), C_1 is the capacitance to ground on the victim net, and C_2 is the coupling capacitance to the perpetrator. R_L, the *node impedance* of the evaluation node, is the effective resistance

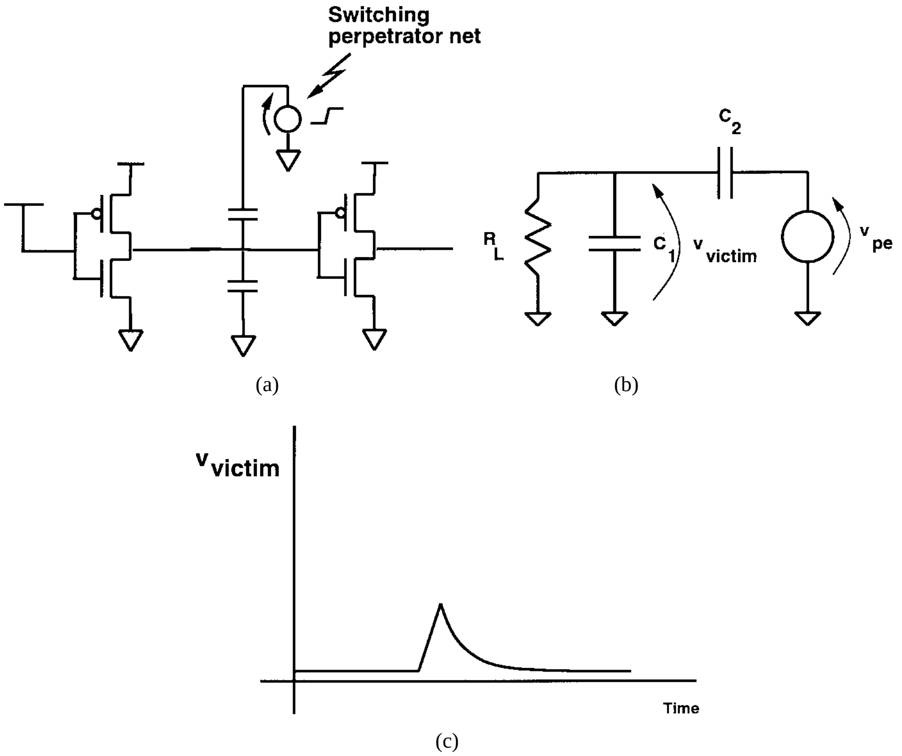


Fig. 3. Capacitive coupling noise: (a) coupling onto an evaluation node between two inverters, (b) simplified equivalent circuit, and (c) pulse coupling noise waveform.

trying to hold the node quiet (in this case, to ground). If the perpetrator is a saturate ramp with slew $m = V_{DD}/t_r$ beginning at $t = 0$, then the noise appearing on the victim net is given by:

$$v_{\text{victim}} = \begin{cases} mR_LC_2\left(1-\exp\left(-\frac{t}{(R_L(C_1+C_2)}\right)\right) \\ \quad \text{for } 0 \le t \le t_r \\ R_LC_2m\left(1-\exp\left(-\frac{t_r}{R_L(C_1+C_2)}\right)\right) \\ \quad \cdot\exp\left(-\frac{t-t_r}{R_L(C_1+C_2)}\right) \\ \quad \text{for } t > t_r. \end{cases} \qquad (1)$$

The resulting noise has the form of a pulse [Fig. 3(c)]. Its leading edge is determined by the switching slew (t_r) on the perpetrator net [assuming $C_1 > C_2$ and $t_r < R_L(C_1 + C_2)$], and its trailing edge is determined by the time constant $\tau_{\text{restoring}} = R_L(C_1 + C_2)$, which we refer to as the restoring time constant. In Section V, we will describe the use of more sophisticated reduced-order models to calculate coupled noise for complex RC interconnect networks. The noise in this case will still have the qualitative pulse-like behavior of Fig. 3(c).

In addition to noise produced by coupling in the interconnect, noise can also be propagated onto an evaluation node from a driving gate, injected by charge-redistribution effects onto the output of a driving gate, or injected by feedback device capacitance onto the input of a receiving gate.

B. Propagated Noise

When V_L noise appears on the input of a CMOS inverter, as in Fig. 4(a), for example, n-FET $M1$ turns on and tries to bring down the output voltage. This action is fought by p-FET $M2$ which continues to hold the output high. Depending on the relative strengths of $M2$ and $M1$, V_H noise is propagated to the output. Skewed-evaluate circuits are more sensitive to noise at their inputs than circuits with balanced rise and fall times. In particular, nodes with a weakened pull-up are more likely to have propagated V_H or V_H^* noise, while nodes with a weakened pull-down are more likely to have propagated V_L or V_L^* noise.

Dynamic circuits, such as the domino AND gate of Fig. 4(b), are an extreme form of skewed-evaluate circuit in which the evaluation transitions are unchallenged. When the clock is zero (the precharge phase), the node D is charged to V_{DD} and the output node Y carries a logic zero. When the clock goes to one (the evaluate phase), and if either A or B is still zero, node D will float with no dc path to ground. Let us consider the case in which B goes high during the evaluate phase, but A is still nominally zero and D is floating [see the voltage waveforms in the inset of Fig. 4(b)]. Because there is nothing fighting to keep node D high V_L noise on A comparable or greater than the n-FET threshold voltage easily propagates to D as V_H noise. We also note that node D is very sensitive to coupled noise for the same reason. One can bolster this gate's noise immunity by including, for example, a (usually weak) p-FET half-latch device as shown in Fig. 4(c). The half-latch device actively fights to keep the dynamic node charged to V_{DD} in the presence of noise. This device, however, degrades performance because it also fights evaluation of the gate. Improved noise immunity almost always comes at a cost in performance or power. With the half-latch, node D is a *weakly-static node* rather than a dynamic node. Static noise

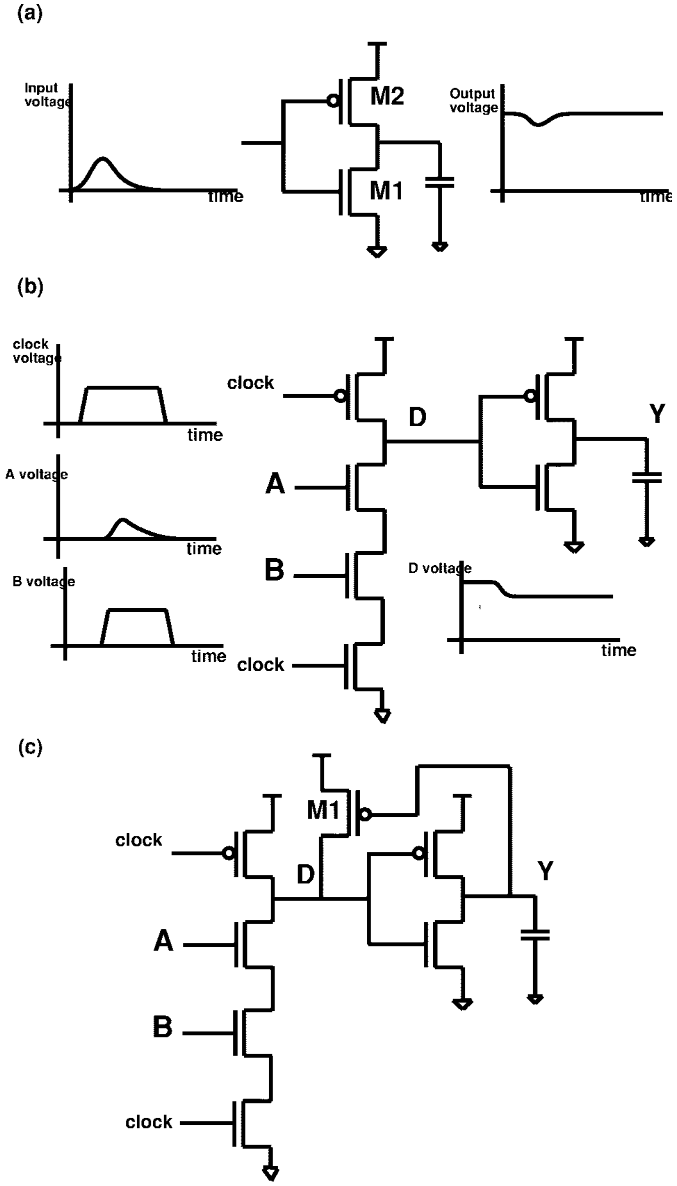
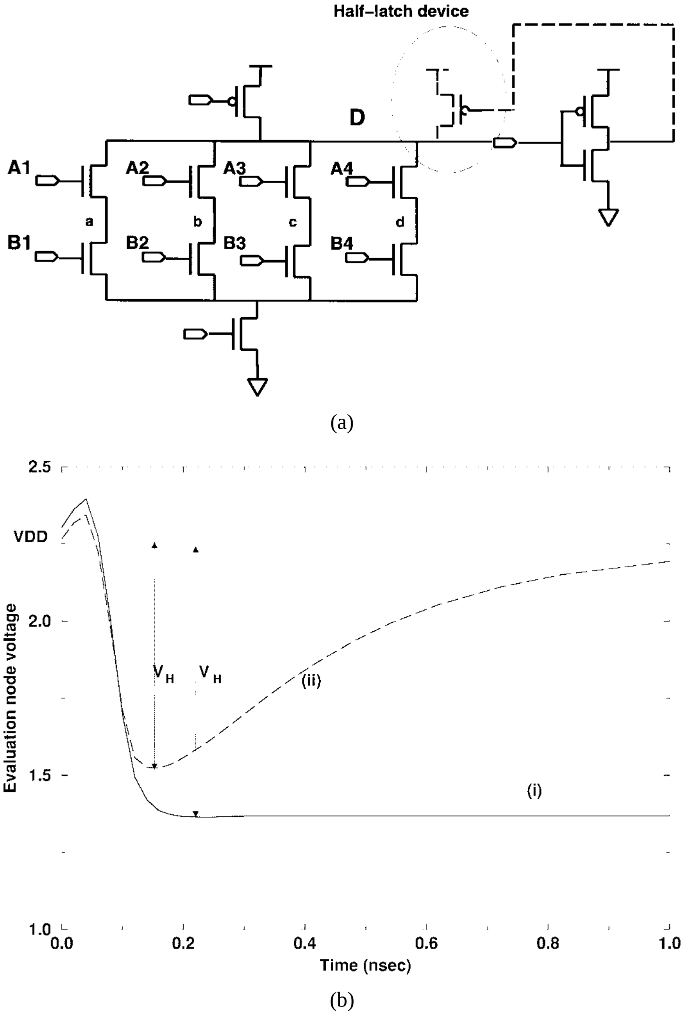


Fig. 4. Circuit noise propagation for (a) a static inverter and (b) a two-input domino AND gate. (c) Adding the half-latch device $M1$ improves the noise immunity of the gate at a performance cost.

analysis techniques must also consider the fact that dynamic and weakly-static nodes are sensitive to subthreshold leakage currents from nominally off devices even in the absence of input noise. Other sources of leakage, such as stray minority carriers in the substrate due to bootstrap noise or ionizing radiation, are described in [4].

C. Charge-Sharing Noise

In addition to noise propagating through a gate, the switching of one net can introduce noise on another by charge-redistribution effects. These effects are most pronounced in skewed-evaluate circuits and are caused by charge sharing between the output node and internal nodes of a pull-up or pull-down stack. In the example in Fig. 5(a), the node D is initially precharged to V_{DD}. Let $B1$ to $B4$ be zero and let $A1$ to $A4$ switch to one. This causes charge sharing between the dynamic node and the internal nodes a, b, c, and d, injecting V_H noise onto D. Graph i) is the noise on D in the absence of the weak half-latch device, while graph ii) shows the noise with the restoring half-latch present. In case ii), the waveform associated with charge-sharing has the same pulse feature as capacitive-coupling noise (cf. Fig. 3). In the absence of the half-latch device, node D is dynamic and never recovers from the charge-sharing noise event. The small peak that is observed at $t = 0.05$ ns is due to the feedback device capacitance coupling between the switching nets $A1$ through $A4$ and node D and is sometimes referred to as Miller noise. This coupling through feedback devices can also produce noise on the input of a receiving gate.



(a)

(b)

Fig. 5. Charge-sharing noise: (a) typical circuit in which node D is a dynamic (in the absence of the half-latch device) or weakly-static node susceptible to charge-sharing noise and (b) V_H noise appearing on node D due to the switching of $A1$, $A2$, $A3$, and $A4$ from zero to one. $B1$, $B2$, $B3$, and $B4$ are zero.

D. Noise Through Device Feedback Capacitance

Consider the example of Fig. 6. The switching of node B results in node C switching from high to low. This couples V_H noise onto node A through the gate-to-source capacitance of device $M1$. If A fans out subsequently to circuits potentially sensitive to V_H noise, functional failure could result.

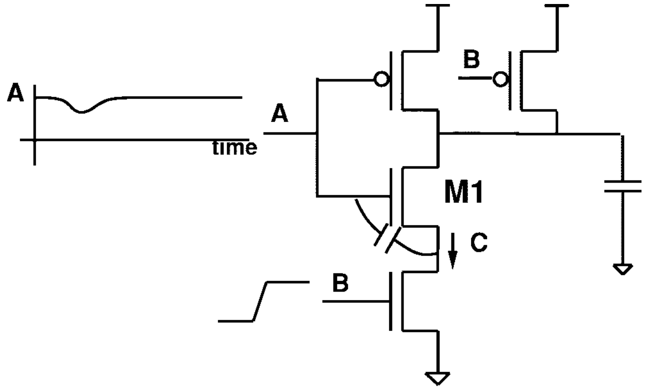


Fig. 6. The switching of node B results in V_H noise on node A due to back-coupling through the gate-to-source capacitance of device $M1$.

E. Substrate and Power-Supply Noise

Switching signals can also introduce noise by means of coupling through the substrate and power supply. Substrate noise results from the fact that the substrate and n-wells are capacitively coupled to device nodes across reverse-biased pn junctions. When devices switch, transient variations in the substrate and n-well bias can occur, which produce threshold voltage shifts by means of the body effect. Power supply noise, on the other hand, appears on the on-chip power and ground distribution network. There are two components to power-supply noise. There are variations in the dc power supply and ground levels due to the average current demands of the chip being drawn through the resistance of the power and ground distribution network (referred to as IR drop). In addition, there is delta-I noise, produced by the simultaneous switching of off-chip drivers and internal circuits, usually synchronized with clock activity. This sudden demand for current causes periodic variations in the supply and ground rails if the current must be supplied through inductance of the chip-package connection. On-chip decoupling capacitance, due to nonswitching circuits, n-well capacitance, or explicit thin-oxide capacitors, provides a transient source of charge that can reduce delta-I noise. In practice, a well-designed on-chip power distribution based on $C4$ technology [10] is sufficiently rigid that the delta-I variations dominate the dc IR drop. We will not consider techniques to calculate substrate [2] or power-supply noise in this paper [6], [11]. Instead, we presume that power and ground variations are bounded by specified dc levels in both the power supply and substrate.

III. Noise Stability as a Metric for Noise Immunity

One traditionally analyzes noise in analog circuits by adding noise generators for each possible physical noise source to the complete small-signal equivalent circuit. These noise generators are usually in the form of mean-square voltages or currents. By contrast, the highly nonlinear operation of digital circuits and the more deterministic nature of man-made noise sources requires an entirely different kind of analysis and verification metric.

Since the publication of the original paper on static noise margins by Hill [12], there have been several papers dealing with the static and dynamic noise margins of logic circuits [13], [14] and memory cells [15]. Instead of using this work as a starting point, we choose to begin at first principles.

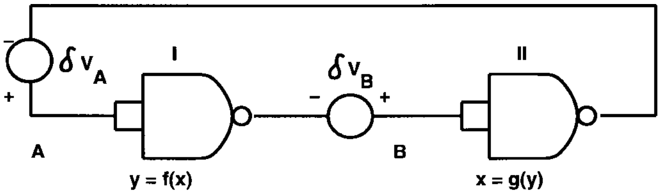


Fig. 7. Latch circuit with dc series-voltage noise sources, δV_A and δV_B.

A. Essential Stability

To guarantee that a digital integrated circuit will function, we must verify that latching structures that hold state do not falsely switch in the presence of noise. Latches can be either static, bistable, positive-feedback configurations of restoring logic gates or dynamic nodes acting as latches, storing state by virtue of the charge on an evaluation node. The act of switching a latch defined by a positive-feedback configuration of restoring logic gates involves making the circuit unstable. Therefore, we refer to the requirement that a latch not be driven unstable by noise as the *essential stability requirement.* Essential stability is the necessary and sufficient condition for the functionality of a digital circuit. Reference [4] presents several examples of essential stability violations.

B. Noise Stability

To verify functionality of a digital circuit, therefore, one could choose to verify the essential stability condition at each latch. With this purpose initially in mind, consider a latch consisting of a bistable feedback configuration of restoring logic gates as shown in Fig. 7. Let x and y be the voltages on nodes A and B, respectively. f and g are the transfer functions of gates I and II, i.e., $y = f(x)$ and $x = g(y)$. The latch will be stable in the presence of the series-voltage dc noise sources (δV_A and δV_B) on evaluation nodes A and B, if at the bias point determined by these sources [13], [14]

$$\left|\frac{\partial f}{\partial x}\frac{\partial g}{\partial y}\right| < 1. \tag{2}$$

That is, the magnitude of the loop gain is less than one. This condition certainly holds in the case that

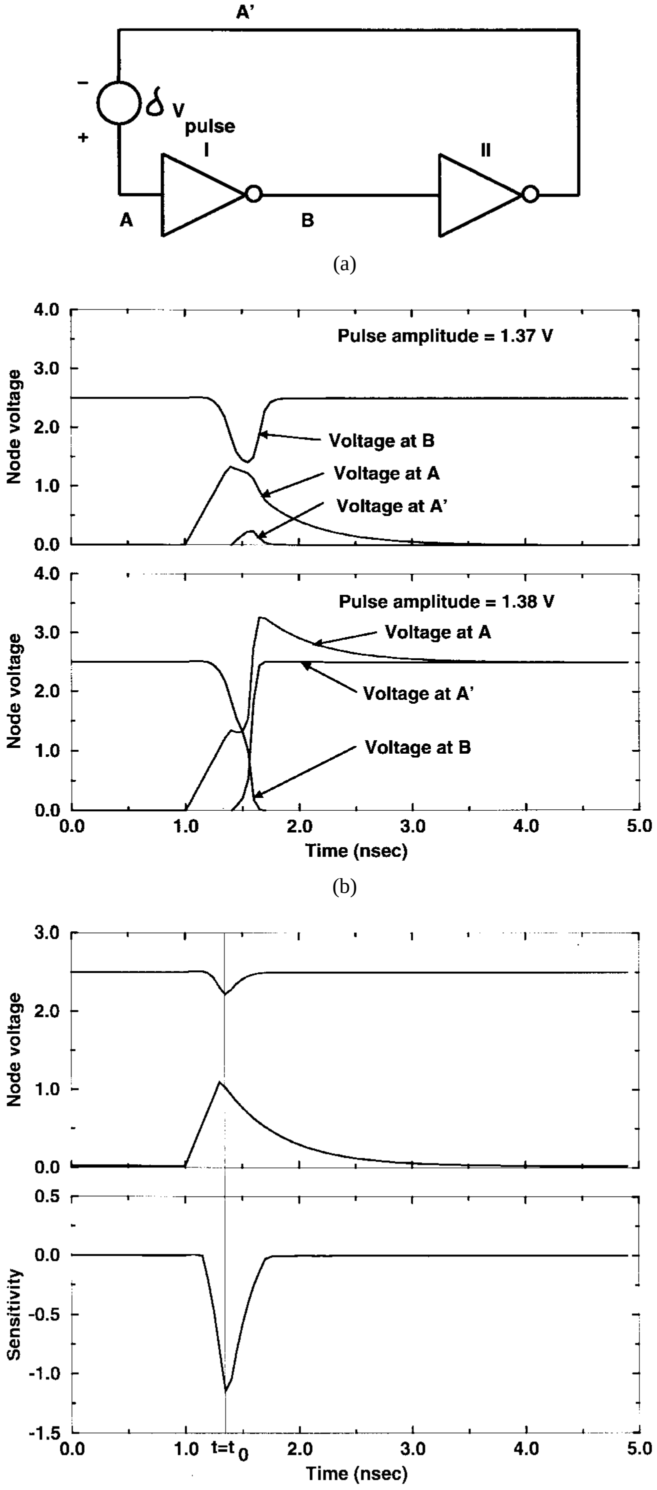
$$|\partial f/\partial x| < 1 \quad \text{and} \quad |\partial g/\partial y| < 1. \tag{3}$$

This stronger condition actually implies additionally that $\delta V_A + \delta V_B$ is maximum [14]. If (3) is applied to every restoring logic gate in the circuit, it is never possible for any positive feedback configuration to switch in the presence of dc noise. This is the condition which is traditionally used to define the worst case static noise margins (or simply static noise margins) [13], [16].

DC noise margins, however, as defined by (3) are much too conservative to apply against the magnitude of pulse noise sources, such as those produced by coupling or charge-sharing, because they fail to consider the fact that logic gates act as low-pass filters. Pulse-noise amplitudes are allowed to be higher than static noise margins would allow, depending on the shape of the pulse. These *dynamic* noise margins are very dependent on the exact time-domain characteristics of the pulse noise [17]. Reference [18] introduces the idea of noise tolerance to define the point at which a symmetric noise pulse shows amplification to the output. The problem with this approach is that it limits the time-domain characteristics of noise to symmetric pulses, which rarely characterize the real behavior of on-chip noise. Seeking a more general metric, we cast the *noise stability* condition, sufficient to ensure functionality, as follows: ***Every restoring logic gate, when acted upon by a noise stimulus, must have a time-domain dc-noise sensitivity that is always less than one.*** The noise stimulus acts to bias the gate, while the dc-noise sensitivity examines the subsequent amplification of additional fluctuations of the lowest possible frequency content (i.e., purely dc).

We consider this condition in more detail for the circuit of Fig. 8(a). In this case, we inject pulse noise onto the series-voltage noise source, δV_{pulse}. The latch is initially in the state in which node A is low and node B is high with a 2.5-V supply. In Fig. 8(b), we show the behavior of the latch when the peak noise amplitude of δV_{pulse} is 1.37 V and when the peak noise amplitude is 1.38 V. In the first case (top graph), the noise is tolerated and the latch does not switch. In the second case, the latch is made unstable and switches, an essential stability violation. In Fig. 8(c), we show how this failure would have been detected by the noise stability check on gate I. The top graph in Fig. 8(c) shows the input and output waveforms for a pulse amplitude of 1.1 V applied to the input of gate I. The bottom graph shows the time-domain dc-noise sensitivity. At time t_o, this sensitivity exceeds unity magnitude. Therefore, at this noise pulse amplitude, the gate is at the threshold of a noise stability violation.[1] The fact that the latch can actually tolerate an additional 280 mV of pulse noise before switching is indicative of the conservatism in the noise stability approach. Because gate II is "subunity-biased," more noise can be tolerated on gate I. This margin is not significant in practice for bistable latch circuits because once a restoring logic gate is biased by noise beyond the unity-sensitivity threshold, the magnitude of the sensitivity rapidly increases. The main source of conservatism in the noise-stability metric comes in applying this test at every restoring logic gate rather than only at latches. We do this to localize the noise failures within a gate or two of the offending noise sources. In practice, noise stability violations, even when they would not result in an essential instability, represent severe design weaknesses which should be corrected.

Noise stability must be verified at the most aggressive conditions under which the chip must be functional—fast process, high temperature, and high nominal voltage. Fast process means faster slews, which generate more coupling noise. Fast process corners also mean shorter channel lengths, which usual results in lower threshold voltages (V_T). Channel length variations can be a significant source of failure due to noise for "fast sorts" if the fast process corner is not used for noise analysis. High temperature means that slews are slower, which generates less noise. However, higher temperature means higher subthreshold currents and more leakage noise, which is generally a much stronger effect. Higher nominal voltages produce faster transitions and higher noise voltage levels relative to V_T.



(a)

(b)

(c)

Fig. 8. Noise stability: (a) a bistable latch circuit containing a pulse noise source, (b) the latch is driven unstable by a pulse amplitude of 1.38 V with the particular choice of pulse shape used in this example, and (c) gate I is noise unstable at a pulse amplitude of 1.1 V.

[1] In the noise graph analysis described in the following section, propagated noise from gate II would be added to the pulse noise, but this noise component is small in this case and neglected for clarity.

IV. Static Noise Analysis

To check an entire digital integrated circuit with tens of millions of transistors for noise stability by means of dynamic simulation is not practical. Instead, static analysis techniques which couple simulations on groups of CCC's with a path trace are used. This technique enables practical checking of noise stability on a chip-wide basis, assuming the worst allowable noise that might be acting in each circuit from all possible noise sources. In this section, we describe the broad assumptions and abstractions that guide *static noise analysis*. In the following section, we describe the additional assumptions and techniques that guided a real implementation used in the design of high-frequency microprocessors at IBM.

Several fundamental assumptions are required to partition the problem of analyzing the noise stability of a digital circuit into small simulations and to combine these simulation results statically:

1) Gate inputs can be replaced by grounded capacitors. This creates a clean partitioning between one CCC and the next and is a technique commonly employed in fast circuit simulation engines [19]. We note that this partitioning associates the interconnect parasitics on an evaluation node with the *driving* CCC. In some cases, large CCC's must be subdivided to contain the run times required for sensitization and simulation analysis.

2) Worst case *sensitization* conditions drive the CCC simulations used for calculating charge-redistribution noise, coupling noise, and propagated noise. By this, we mean how the transistor gates are biased, how the noise stimulus or switching waveform is applied, and the initial voltages on internal nodes. Sensitizations must honor static logic constraints; that is, constraints which are true once all of the logic signals settle. We will describe the conditions that guide this sensitization in more detail below.

3) We assume that the superposition principle applies in adding (in the time domain) circuit noise and interconnect noise sources. For noise sources small enough to satisfy the noise stability requirement, active FET channels (that is, those attempting to hold nodes to their static level) are biased in the triode regions of their current–voltage characteristics, justifying this "linear" assumption. In particular, charge-sharing noise and propagated noise can be calculated on a "single-input" changing basis and superposed with the coupled noise calculations to find the total noise. The sensitization producing the largest amplitude output noise is used. Noise sources can only be combined when the sensitization conditions are consistent. We choose not to include Miller noise effects since they are usually small. The effect of stray minority carriers in the substrate (e.g., as produced by ionizing radiation) are not explicitly considered but could be included as a time-dependent current source on an evaluation node.

4) Worst case temporal relationships are defined by superposing the peak responses of the charge-redistribution, coupling, and propagated noise for each allowable noise (V_L, V_H, V_L^*, V_H^*) type. (One might argue that a superposition producing a lower amplitude but wider pulse response might be worse in some cases than the larger amplitude noise. While we do not rule out this possibility, we have found that in practice the larger amplitude superposition is almost always the most destabilizing to receiving circuits.) When timing information is known, it can be used to reduce pessimism in combining noise sources by disallowing the simultaneous switching of signals with nonoverlapping arrival time windows. In addition, logic constraints which are associated with hazard-free logic can be used to disallow simultaneous switching events. This will be described in more detail in the discussion below.

5) A noise stability check as described in Section III is performed across every restoring logic gate in the design. Noise stability violations are assumed to be a sufficient condition for finding the circuit to be nonfunctional.

6) Substrate and power-supply integrity analyses are performed independently and are generally characterized for static noise analysis by dc bounds on the local power and ground variations. In calculating propagated noise, collapsed rails are used, characterized by dc values $V_{DD}^{\min}$ and $\mathrm{Gnd}^{\max}$. In doing a noise stability check, expanded rails are used, characterized by dc values $V_{DD}^{\max}$ and $\mathrm{Gnd}^{\min}$.

7) Drivers on switching perpetrator nets (which we refer to as *secondary nets*) are modeled as ideal voltage sources. This presumes that the noise-on-delay effect has been handled elsewhere.

8) In the case that the circuit contains feedback (as in a latch circuit), the feedback loop is broken at a restoring logic gate. Two approaches can be used at the cut points. The simplest is to assume the worst possible dc noise that can be propagated without producing a stability violation in the "broken" gate. To determine the magnitude of this dc *noise-limited* propagated noise, the subunity gain criterion is applied to a dc voltage transfer characteristic with supplies defined by $V_{DD}^{\min}$ and $\mathrm{Gnd}^{\max}$. The second approach is an iterative one, in which initially no noise is assumed to be propagated across the broken gate. Once the input noise on the broken gate is calculated, this is then propagated to the output and the process repeated until convergence.

9) Noise-limited propagation is used for restoring logic gates which have a stability violation. This allows the noise analysis to continue, despite the violation, and places the burden on fixing the noise problem on the circuits driving the violating gate.

10) Full-rail signaling is assumed. For example, the current approach to static noise analysis does not handle partial-voltage-swing differential circuits (e.g., $V_{DD}/2$-precharging of bit lines and sense-amplifier detection in SRAM's).

The key abstraction in static noise analysis is the noise graph, a directed graph containing all of the circuit's evaluation nodes connected by segments that move and transform noise.

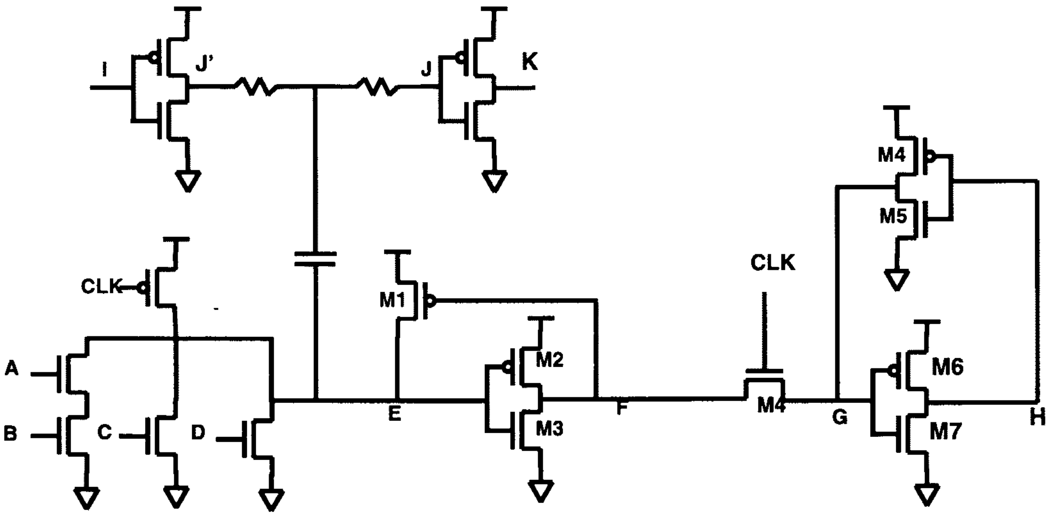
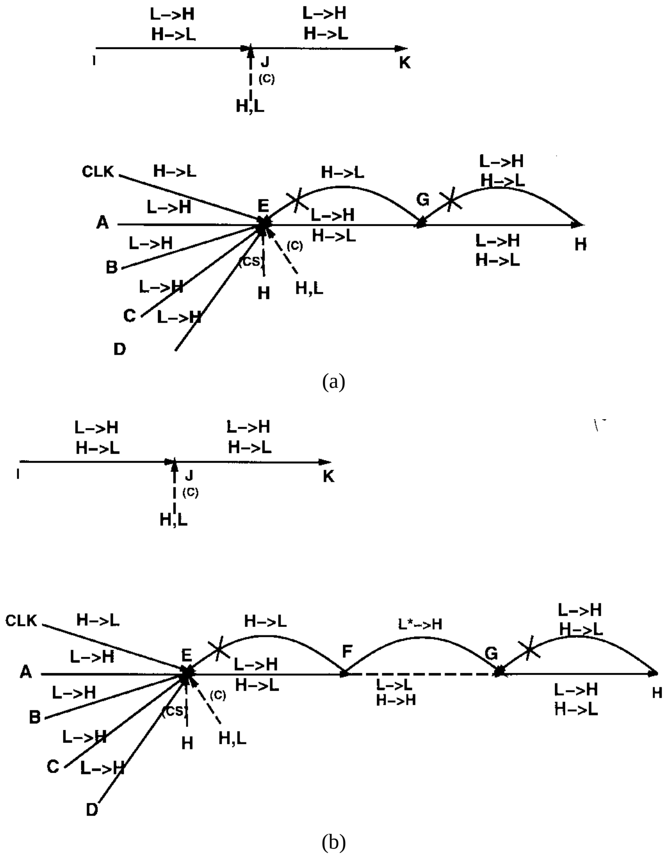


Fig. 9. Example circuit for noise analysis consisting of a domino gate driving a latch. Node E is capacitively coupled to another net.

In many ways, this graph is analogous to the timing graph used in static timing analysis. An example circuit is shown in Fig. 9 and two possible noise graph representations are given in Fig. 10. Fig. 10(b) differs from Fig. 10(a) in that the pass transistor has been partitioned from its associated CCC by the introduction of the evaluation node F. (The additional assumptions associated with analyzing only a part of a CCC will be described later in this section.) There are three types of segments in a noise graph: restoring segments, propagate segments, and node-injection segments. *Restoring segments* cross gates that at some dc bias point have a small-signal gain greater than one. Noise is propagated across restoring segments; in addition, a noise stability check must also be performed. *Propagate segments* [e.g., the dashed line joining nodes F and G in Fig. 10(b)] connect nodes, between which there is subunity gain at all dc bias points. Noise-stability checking is not required across propagate segments. Each restoring and propagate segment in the noise graph is labeled by the type of noise propagated by the segment. For example $L \rightarrow H$ indicates that the segment propagates V_L noise and transforms it into V_H noise. The *node-injection segments* (dashed lines in Fig. 10 that are not sourced by nodes) can introduce noise directly onto an evaluation node, superposing with the propagated noise. Coupled interconnect noise, denoted by (C), and charge-sharing noise, denoted by (CS), are both modeled as node-injection segments. Once the noise graph is constructed, the loops of the graph are broken and the graph is topologically sorted for traversal. In Fig. 10(a), the segment from H to F is snipped to break the loop as is the segment associated with the half-latch from F to E. Noise is then propagated across each of the "loop snips" (Assumption 8). The graph is then searched in a breadth-first fashion to propagate noise through the network, and in the case of restoring segments, to perform the sensitivity tests required to ensure noise stability.



(a)

(b)

Fig. 10. Noise graphs for the circuit of Fig. 9 during the evaluate phase of the domino gate (Input CLK is high). (a) graph in the case of strict CCC partitioning and (b) graph in which the pass transistor $M4$ is partitioned by creating the evaluation node F.

In general, transistor path-based functional extraction [20] guides three main types of sensitizations (Assumption 2): sensitization for coupled noise calculation on the output node of a CCC, sensitization for noise stability and propagated noise calculation from a given input, and sensitization for charge-redistribution noise calculation from a switching waveform on a given input. Transistor direction setting [21], [22] facilitates the path-function extraction in cases in which the required path function is to V_{DD} or ground. Allowable sensitizations are determined by the Boolean satisfiability of constraint relations determined by this functional extraction. Logic conditions between the input variables (denoted as f_{input}), when they exist, must be included in these constraint relations. In the case of multiphase design, time-sliced sensitization analysis must be used in general; that is, sensitization must be considered separately at each clock phase and the results of the analysis of one phase applied to the next. Logic constraints must also be generally specified for each time slice. Clock signals in this context are simply constants specified on such a time-sliced basis. General time-sliced analysis is beyond the scope of this paper, although a highly simplified approach for two-phase dynamic logic is applied to Harmony and described in Section V.

The analysis of each CCC involves calculating, through transistor-level simulations, the noise appearing at each CCC output and verifying the stability of the noise waveforms appearing at each input. The noise calculation begins by establishing the dc voltages (or base levels) associated with logic high and logic low. These can differ from the rails by a threshold-voltage drop in the case of noncomplementary pass gates, for example. As part of this analysis, input combinations that cause the output of a CCC, O, to float (have no path to V_{DD} or ground) or collide (have paths to both V_{DD} and ground) are also examined. Both conditions should be reportable to the user since they sometimes represent unintended circuit behavior. Collision cases must be individually verified to determine if they unambiguously resolve to a logic high or logic low (e.g., ratioed logic has valid ratioing). In the case of floating nodes, a dc base level must be asserted at the output as an initial value (e.g., as might result from a previous phase precharge level). Subthreshold leakage in the case of dynamic or weakly-static nodes must also be considered as part of the dc base-level analysis. This leakage is allowed to act for a clock-period-dependent period of time to determine the final degraded base level.

Having established the base levels of logic high and low, we now consider the possible ways noise can upset this voltage, beginning with coupled noise. We define a path function $f_{P_{i,j}}$ as the logical condition for the channel path from i to j to conduct. To sensitize for noise appearing on O due to capacitive coupling to a given node, D, in the CCC, we establish logic constraint relations depending on the type of noise propagating from D to O. In particular, let us consider the sensitizations that allow V_L noise to appear on O due to capacitive coupling to D. In this case, the V_L noise at O is produced by a perpetrator net switching from ground to V_{DD}. There are two possible constraint relations. The first

$$f_{P_{O,\text{Gnd}}} \cdot f_{P_{O,D}} \cdot f_{\text{input}} \tag{4}$$

allows V_L noise on D to propagate to O, while the second

$$f_{P_{O,\text{Gnd}}} \cdot \overline{f_{P_{O,D}}} \cdot f_{P_{D,V_{DD}}} \cdot \overline{f_{P_{D,\text{Gnd}}}} \cdot f_{\text{input}} \tag{5}$$

allows V_H^* noise on D to propagate to O as V_L noise. All input sensitizations (as determined by binary-decision diagram analysis [23]) that satisfy one of these two constraints can inject V_L noise onto node O due to coupling on node D. In a similar way, the two possible constraint relations that allow V_H noise to appear on the output O are given by

$$f_{P_{O,V_{DD}}} \cdot f_{P_{O,D}} \cdot f_{\text{input}} \tag{6}$$

and

$$f_{P_{O,V_{DD}}} \cdot \overline{f_{P_{O,D}}} \cdot f_{P_{D,V_{DD}}} \cdot \overline{f_{P_{O,\text{Gnd}}}} \cdot f_{\text{input}}. \tag{7}$$

In this case, the secondary net capacitively coupled to D is switching from ground to V_{DD}. As an example, consider the circuit shown in Fig. 11 with the static constraints that C and D must be complementary and O and E must be complementary. To calculate V_H noise on O due to the switching of secondary driver F from high to low, there are several valid sensitizations (1–7 shown in Table I). A "0" or "1" entry in the table indicates that the given input is held at nominal ground or V_{DD}, respectively, during the simulation. "Rise" and "fall" denote switching waveforms. For sensitizations 1–4, the pass page is off and, therefore, all four of these sensitizations should result in nearly the same noise at O. For sensitizations 5–7, the pass gate is on and the amount of coupling noise will depend on the strength of the path to V_{DD}.

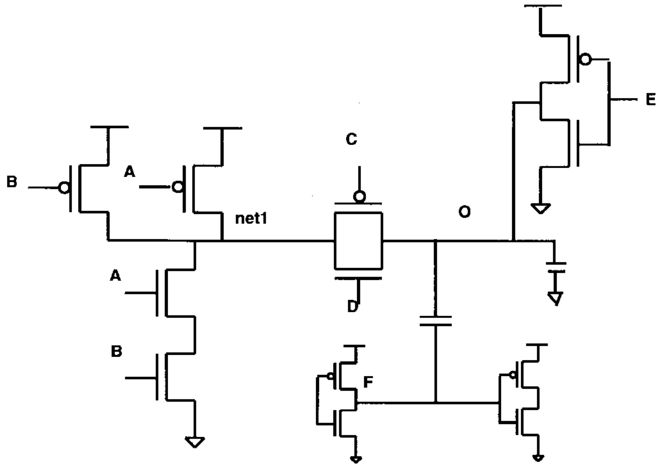


Fig. 11. Static NAND gate driving a pass gate latch. The gate input of one of the inverters of the latch has been replaced by a linear capacitor according to Assumption 2. We use this example to describe sensitization conditions.

TABLE I

"Single-Noise-Source" Sensitizations for V_H Noise at Node O for the Example of Fig. 10. A "0" or "1" Entry in the Table Indicates that the Given Input Is Held at Nominal Ground or V_{DD}. "Rise" and "Fall" Denote Switching Waveforms. The "X" Indicates that the Given Variable Can Have Either "0" or "1" Value

sensitization	A	B	C	D	E	F
1	0	0	0	1	0	fall
2	0	1	0	1	0	fall
3	1	0	0	1	0	fall
4	1	1	0	1	0	fall
5	0	0	1	0	0	fall
6	0	1	1	0	0	fall
7	1	0	1	0	0	fall
8	V_L	1	0	1	0	X
9	1	V_L	0	1	0	X
10	1	1	V_L	1	0	X
11	rise	1	0	1	0	X

To determine the noise propagated from a CCC input to CCC output, the target input is stimulated by the noise propagated to that point in the noise-graph traversal. n-FET's are sensitive to V_L noise on their gates while p-FET's are sensitive to V_H noise. Let the input receiving the noise be I. Let D_1 denote the internal node on the output side of the target

FET connected to I, and let D_2 denote the other channel node of this target FET. For V_H noise to propagate to the output the constraint relation is

$$\begin{aligned} &f_{P_{O,V_{DD}}}(I=0) \cdot f_{\text{input}}(I=0) \cdot f_{P_{D_1,O}}(I=0) \\ &\quad \cdot f_{P_{D_2,\text{Gnd}}}(I=0) \cdot f_{P_{O,\text{Gnd}}}(I=1) + f_{P_{O,V_{DD}}}(I=1) \\ &\quad \cdot f_{\text{input}}(I=1) \cdot f_{P_{D_1,O}}(I=1) \cdot f_{P_{D_2,\text{Gnd}}}(I=1) \\ &\quad \cdot f_{P_{O,\text{Gnd}}}(I=0) \end{aligned} \tag{8}$$

while for V_L noise to propagate to the output, the logical condition is given by

$$\begin{aligned} &f_{P_{O,\text{Gnd}}}(I=1) \cdot f_{\text{input}}(I=1) \cdot f_{P_{D_1,O}}(I=1) \\ &\quad \cdot f_{P_{D_2,V_{DD}}}(I=1) \cdot f_{P_{O,V_{DD}}}(I=0) + f_{P_{O,\text{Gnd}}}(I=0) \\ &\quad \cdot f_{\text{input}}(I=0) \cdot f_{P_{D_1,O}}(I=0) \cdot f_{P_{D_2,V_{DD}}}(I=0) \\ &\quad \cdot f_{P_{O,V_{DD}}}(I=1). \end{aligned} \tag{9}$$

The notation $f_{P_{O,V_{DD}}}(I=0)$, for example. indicates that the path function from O to ground is evaluated with input I at zero. Note that these sensitization conditions explicitly check that a full transition on the target input would switch the output. In the example of Fig. 11, to propagate V_L noise from A to O as V_H noise, B and D must be set to one, and A, C, and E must be set to zero. This corresponds to sensitization 8 in Table I. The "X" in the table indicates that the given variable can have either "0" or "1" value, as is the case for F in this sensitization. V_L indicates that the given input is stimulated by V_L noise. Sensitizations 9 and 10 in Table I correspond to V_L noise propagated from inputs B and C, respectively.

For charge-sharing noise calculation, a FET gate is receiving a switching signal. For waveforms switching from ground to V_{DD}, the target FET will be an n-FET. For waveforms switching from V_{DD} to ground, the target FET will be a p-FET. Let the switching input be I. Let D_1 denote one of the internal nodes of the target FET connected to I, and let D_2 denote the other channel node. For V_L noise at the output, let

$$\begin{aligned} f_1 &= f_{P_{O,\text{Gnd}}} \cdot f_{P_{D_1,O}} \cdot \overline{f_{P_{D_2,V_{DD}}}} \cdot \overline{f_{P_{D_2,Gnd}}} \cdot f_{O,D_2} \\ f_2 &= f_{P_{O,\text{Gnd}}} \cdot f_{P_{D_2,O}} \cdot \overline{f_{P_{D_1,V_{DD}}}} \cdot \overline{f_{P_{D_1,\text{Gnd}}}} \cdot f_{O,D_1}. \end{aligned}$$

If the target FET is an n-FET, the logical constraint for V_L noise is

$$\begin{aligned} &\big(f_1(I=1) + f_2(I=1)\big) \cdot f_{P_{O,\text{Gnd}}}(I=0) \\ &\quad \cdot \big(f_{\text{input}}(I=0) + f_{\text{input}}(I=1)\big) \end{aligned}$$

while if the target FET is an n-FET, then the constraint is

$$\begin{aligned} &\big(f_1(I=0) + f_2(I=0)\big) \cdot f_{P_{O,\text{Gnd}}}(I=1) \\ &\quad \cdot \big(f_{\text{input}}(I=0) + f_{\text{input}}(I=1)\big). \end{aligned}$$

Similar equations follow for V_H noise. Note that these sensitization conditions explicitly check that a transition on the output will not be produced by the full transition of the target input. Also note that the input constraints are smoothed with respect to the switching input since the switching input does not have to satisfy static logic constraints. In the example of Fig. 11, if the NAND gate is very skewed in favor of the pull-down, then V_H charge-sharing noise can be introduced at O by the switching of input A from low to high. In this case, B and D are one, while E and C are zero. This corresponds to sensitization 11 in Table I.

TABLE II
COMPLETE SENSITIZATION POSSIBILITIES FOR V_H NOISE ON NODE O OF FIG. 11. A "0" OR "1" ENTRY INDICATES THAT THE GIVEN INPUT IS HELD AT NOMINAL V_{DD}. V_L INDICATES THAT THE GIVEN INPUT IS STIMULATED BY V_L NOISE, WHILE "RISE" AND "FALL" DENOTE SWITCHING WAVEFORMS

sensitization	A	B	C	D	E	F
1	0	0	0	1	0	fall
2	0	1	0	1	0	fall
3	1	0	0	1	0	fall
4	1	1	V_L	1	0	fall
5	rise	1	0	1	0	fall
6	0	0	1	0	0	fall
7	V_L	1	0	1	0	fall
8	1	V_L	0	1	0	fall

In general, to find the noise appearing at the output of a given CCC (Assumption 3), we must find the combined sensitization producing the largest amplitude output noise for each noise type (V_L or V_H). In the example of Fig. 11, there are eight possible combined sensitizations for V_H noise appearing on node O that come from the superposition of entries in Table II (as enumerated in Table I). Sensitizations 4 and 5 may be additionally combinable if it is known that the rising transition on A occurs before the V_L pulse on C. Static noise analysis picks the sensitization producing the worst peak noise at the output. In addition to calculating the worst case noise that can appear on each output, for each noise appearing at a CCC input, we calculate the worst dc-noise time-domain sensitivity for all the possible patterns that satisfy (8) or (9). Establishing that this sensitivity is always less than unity in magnitude is sufficient to guarantee functionality of the design (Assumption 5).

Two additional types of constraints on the switching signals can be used to further limit noise combinations.

- *Hazard-Free Logic Constraints*: These are logic constraints that apply to signals that are known to be hazard-free. For example, if two hazard-free signals are complementary, then a rising transition on one implies a single falling transition on the other.
- *Timing Orthogonality*: If two signals cannot switch together as a result of static timing analysis, then the simultaneity of these two switching events is precluded in combining noise sources.

In the example of Fig. 11, suppose that we know that A and F satisfy the constraint $A\ xnor F$ and that additionally both A and F are hazard-free. In this case, we know that sensitization 5 in Table II cannot occur because these nets cannot switch in opposite directions. Now, assume that we have no logic constraints on A and F but we know from static timing analysis the earliest and latest arrival times for the rising and falling edges of A and F (i.e., arrival time windows). In sensitization 5, peak superposition of the noise resulting from the switching of A and F implies a relative timing of the rising edge of A to the falling edge of F. These edges must fall within the arrival time windows defined by static timing analysis or sensitization 5 must be disallowed. We will discuss

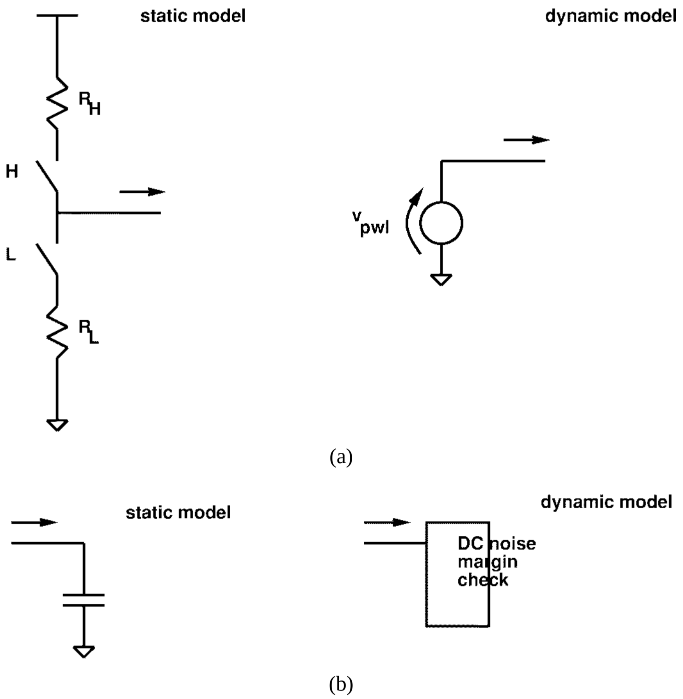


Fig. 12. Port models: (a) noise-input and (b) noise-output.

the details of this *timing orthogonality* analysis in the context of Global Harmony in the following section.

Special effort is applied to correctly modeling the ports or pins of the design. Ports are introduced, for example, when the design is partitioned for hierarchical analysis. Ports are of three types—noise-inputs, noise-outputs, or noise-bidis. Noise-inputs are pins through which noise can be injected; noise-outputs are pins from which noise can be propagated; while noise-bidis can function as both noise-inputs and noise-outputs. Ports of the network must be modeled in two ways, statically to determine how the ports act to hold nets quiet in the presence of noise and dynamically to determine the noise that is propagating in on the given pin or that can be tolerated on a given pin. The static model for an input port, as shown in Fig. 12(a), describes how the port acts to hold a net quiet. R_H and R_L are the node impedances, the effective pull-up and pull-down resistances controlled by the variables H and L, which participate in sensitization along with other input variables. In most cases, H and L are constrained to be mutually exclusive, which precludes floating node and collision conditions. For an output pin, the static model is a capacitor as shown in Fig. 12(b). The dynamic model for noise-input pins consists of a piece-wise-linear voltage source $(v_{\rm pwl})$ connected to the pin. Noise-outputs can have a dynamic model characterized by a dc noise margin check.

V. Harmony

Successful design methodologies incorporate a three-tiered strategy for noise. A set of noise avoidance rules guide circuit and interconnect design. Examples include maximum tolerable skews for static CMOS gates, minimum half-latch strengths, spacing-length routing rules for interconnect, and restrictions on the use of pass-gate latch inputs. These rules are chosen to prevent most noise problems but should not be too restrictive as to create deleterious constraints on area or timing. Static noise analysis is then run on the entire design to find all possible noise failures, in particular those not caught by the design rules. Last, in special cases and because of the inherent conservatism of static noise analysis, some failures flagged by static noise analysis may be allowed after careful circuit simulation. We discuss the risks of bypassing the conservatism in static noise analysis in Section VI.

Harmony is a two-level hierarchical implementation of static noise analysis that was used as part of such a methodology in the design of high-performance CMOS microprocessors within IBM [6]. The Harmony implementation is consistent with a parallel two-level-hierarchical static timing and parasitic extraction flow. The overall architecture is shown in Fig. 13. This two-level hierarchical division is necessary to practically handle the complexity of designs with tens of millions of transistors. The methodology involves identifying groups of 1000–200 000 transistors as *macros*. (In some cases, it is convenient to define macros with as few as 100 transistors.) Macros are individually laid out and floorplanned on the chip. They are timed using static timing analysis and abstracted in delay calculation language (DCL) [24]. Similarly, static noise analysis is performed on each macro (Macro Harmony) and noise abstracts are generated for the global analysis (Global Harmony). In some cases, noise assertions are returned back to Harmony for analysis. In this hierarchical partitioning, the global level consists entirely of the interconnection network between the macros which contains all the long wire runs of the chip. The coupling capacitances between the macro and global levels are handled by treating them as worst case hostile coupling sources in both the Macro Harmony and Global Harmony analyses.

A. Macro Analysis—Macro Harmony

Several methodology-specific assumptions guide the Macro Harmony implementation. Macros are assumed to be sufficiently small that resistance in the interconnect can be ignored; this enables very simple capacitance-only interconnect models. Timing information is not known *within* the macros because the methodology does not include a tight linkage between static timing analysis and static noise analysis at the macro level. As a result, switching signals are bounded by worst case and best-case slews specified as the parameters, $t_{\rm slew}^{\rm max}$ and $t_{\rm slew}^{\rm min}$. To further simplify the transistor-level analysis, we made several other important implementation decisions. In future implementations of static noise analysis, more sophistication can be expected and many of these simplifications can be removed.

In lieu of propagating detailed time-domain waveshapes and performing expensive time-domain superposition, we chose to implement simplified time-domain abstractions for the analog noise waveshapes. This is similar in philosophy to the use of saturate ramps in static noise analysis. In particular, the noise on any node is treated as the superposition of a dc noise and a pulse noise. Pulse noise is characterized by a peak value, a leading slew time given by $t_{\rm slew}^{\rm max}$, and a trailing restoring time constant. In addition to this abstraction, noise propagating through any logic stage is treated as dc at the output; that is

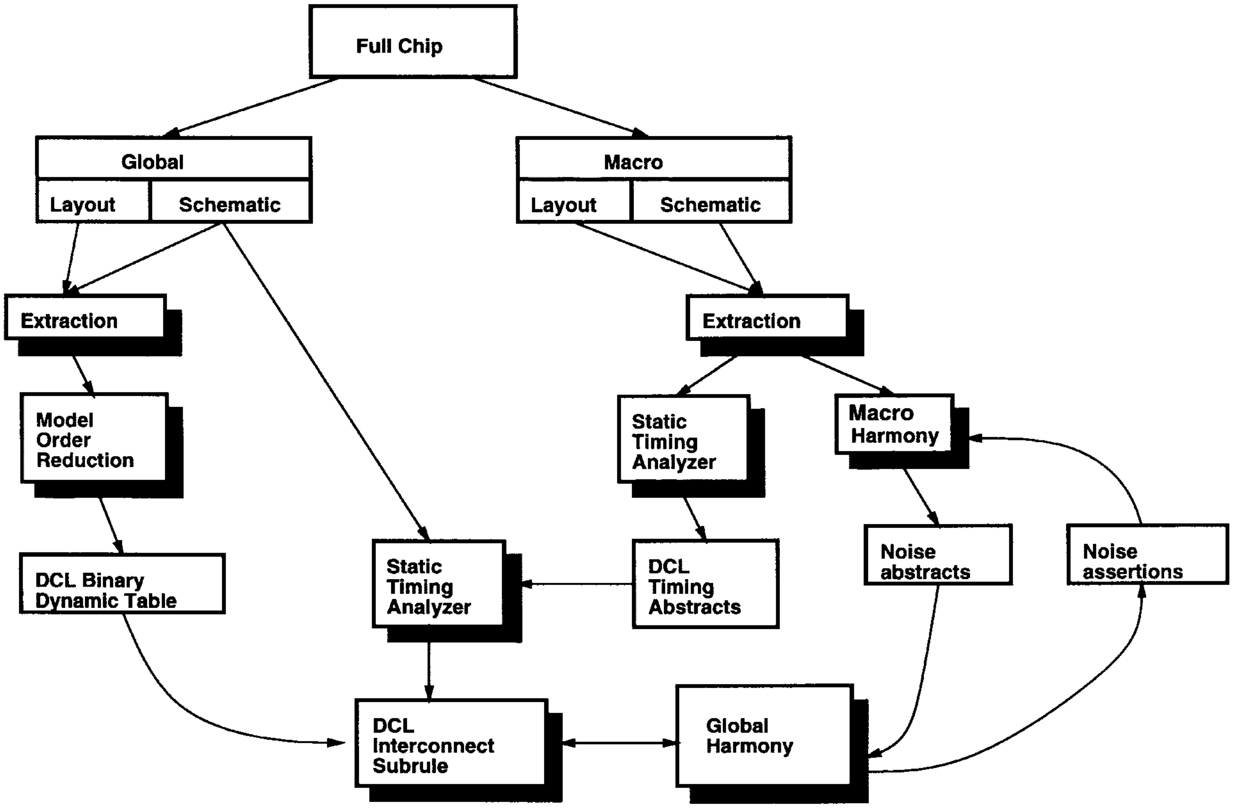


Fig. 13. Harmony architecture for static noise analysis. The hierarchical division used for noise analysis follows exactly that used for static timing analysis.

the peak noise is extended to be a dc value. This simplifies the analysis at the cost of additional pessimism.

Instead of selectively breaking up large CCC's when necessary, we took advantage of the fact that large CCC's are nearly always associated with pass-transistor networks. For pass-transistors, we choose to separate these from their associated CCC's and include them as separate elements in the noise graph, defining additional evaluation nodes to delineate the pass-gate channels. Fig. 10(b) shows the noise graph for the circuit of Fig. 9 in the case that a new evaluation node is created at F to partition the pass gate from F to G. A propagate segment is used for channel conduction as denoted by the dashed line from F to G. It is obvious for this graph that additional noise stability checks accompany this partitioned pass-gate analysis. In particular, there is now a restoring segment from F to G that propagates V_{L^*} noise as V_H noise.

In Macro Harmony, to simplify the analysis, each of the sensitizations for coupled noise, propagated noise, and charge-sharing noise is treated independently. The worst case noise sensitization for each noise type is used, even though these sensitizations may be inconsistent. To limit the number of simulations, very simple heuristics guide determining the worst case sensitizations for each type of noise that satisfy the constraint relations introduced in Section V. Additionally, we reduce the coupled noise calculation to a closed form analytic expression using the *node impedance*, R_H and R_L for each evaluation node. Following (1), pulse noise is characterized by a restoring time constant of $\tau_{\text{restoring}} = R_{H/L}C_{\text{self}}$, where C_{self} is the self-capacitance of the evaluation node, which includes linearized gate and diffusion capacitances. Peak coupled noise produced by a coupling capacitance C_{coupling} to a source switching with slew time $t_{\text{slew}}^{\text{min}}$ is given by

$$R_{H/L}C_{\text{coupling}}\frac{V_{DD}}{t_{\text{slew}}^{\text{min}}}\left(1 - e^{-t_{\text{slew}}^{\text{min}}/(R_{H/L}C_{\text{self}})}\right). \tag{10}$$

We designed Macro Harmony to handle static CMOS, pass-gate logic, and domino, but the techniques of static noise analysis can be easily extended to other circuit styles. In lieu of more complex time-sliced functional analysis, Harmony employs topological recognition of dynamic logic, enabled by a simple clock-propagation algorithm that traces clocks through static CMOS gates. Noise analysis of dynamic logic presumes that the clocks are sensitized for evaluation, rather than precharge; that is, we only choose to calculate noise acting during the evaluate phase of the domino gate.

Two types of reports are generated as a result of the Macro Harmony run. The first reports the noise appearing on each evaluation node in the circuit, classified by noise type. The second reports all noise stability violations. As part of the macro-level static noise analysis, noise abstracts are also generated for macro blocks. These noise abstracts are simply port models, a noise-input model for each macro noise output and a noise-output model for each macro noise input. As part of the creation of the noise-output model for each macro noise input, a dc noise margin is calculated for each relevant noise type using a sensitivity analysis at the first restoring logic gate from the pin. In addition, a noise-limited dc value is propagated from this first restoring logic gate in the Macro Harmony analysis when abstracts are generated. In some cases, this pessimism produces false violations both in the macro and

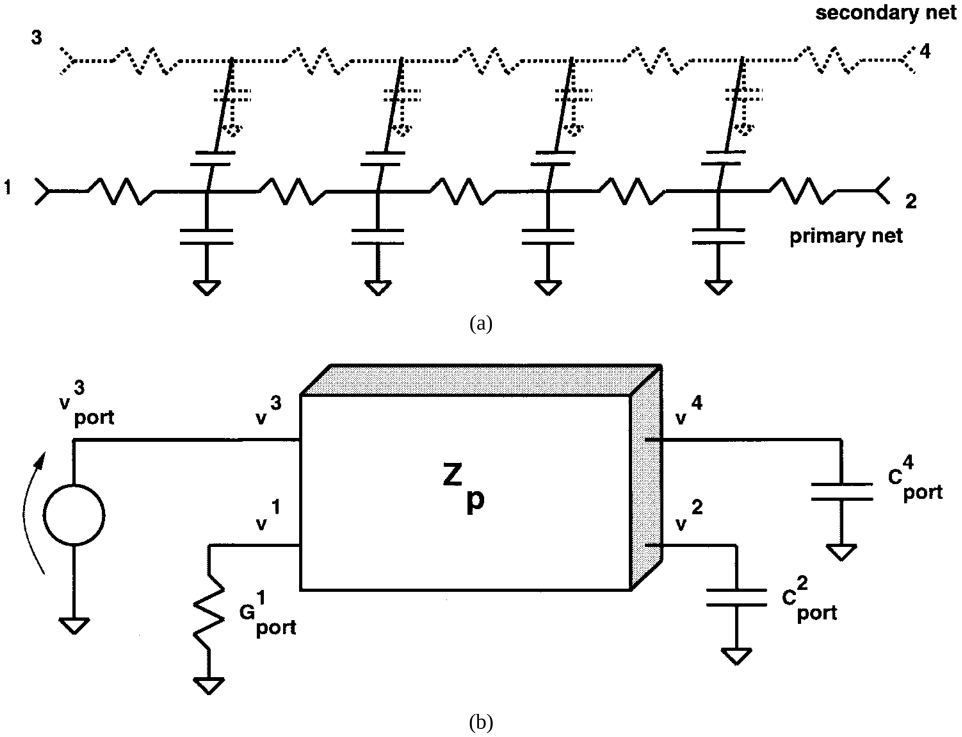


Fig. 14. Multiport modeling of global interconnect: (a) a typical net complex consisting of a primary net coupled (in this case) to a single secondary net and (b) the driver resistances and receiver capacitances are folded into the multiport impedance macromodel.

at the global level. In this case, assertions of the actual pulse noise calculated in Global Harmony can be used as an input to subsequent Macro Harmony runs.

B. Global Analysis—Global Harmony

Once Macro Harmony has been used to analyze each macro block, we must consider all the long interconnect of the chip. The Global Harmony engine is nothing more than a detailed coupled noise calculator, since all the transistor-level analysis is done and abstracted by Macro Harmony. Interconnect resistance is included in the Global Harmony interconnect analysis. In addition, timing information becomes very important in reducing pessimism, since most of the coupled noise is introduced in the global wires connecting the macros. This global detailed timing information is also available in the design methodology in which Global Harmony is employed. The extraction of the global interconnect results in an RC network that is reduced in Global Harmony to a collection of multiport impedance macromodels, one for each net in the design, stored as a DCL binary dynamic table. The reduced-order modeling approach employed in Global Harmony guarantees passive, multiport macromodels with symmetry that allows for efficient storage of the results. Multiport models are used so that the interconnect models remain independent of changes in the macro driver strengths and input pin capacitances. These macromodels are also employed in the static timing analysis of the same design.

The first step in the reduction process is to identify a *net complex* for each global net in the design. The *primary net* of the complex is the net on which we are trying to calculate the noise; that is, the net which should be statically quiet. The complex also includes *secondary nets* of significant coupling to the primary net. To determine which secondary nets to include in a complex, we calculate the ratio $C_{\text{coup}}/C_{\text{self}}$ for each secondary net, where C_{coup} is the total coupling capacitance to the given secondary net and C_{self} is the self capacitance of the primary net in the complex. Secondary nets for which this ratio is below a designated threshold are discarded. Coupling capacitances to discarded secondary nets are treated as capacitors tied to ground. Couplings between the significant secondary nets and nets other than those already in the net complex are grounded. A representative net complex is shown in Fig. 14(a).

Modified nodal analysis (MNA) is used to stamp conductance and capacitance matrices according to the multiinput, multioutput, linear time-invariant differential equations

$$\boldsymbol{C}\dot{\boldsymbol{x}} = -\boldsymbol{G}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{i} \qquad \boldsymbol{v} = \boldsymbol{B}^T\boldsymbol{x}. \tag{11}$$

$\boldsymbol{x}$, $\boldsymbol{v}$, $\boldsymbol{i} \in \mathbb{R}^n$ are the state, output voltage, and input current vectors, respectively. For a system with n nodes and r ports, $\boldsymbol{G}$, $\boldsymbol{C} \in \mathbb{R}^{n \times n}$ are the symmetric, positive semidefinite conductance, and capacitance matrices, respectively. The state vector is ordered so that the first r elements represent the port voltages. With this choice of ordering, the r-by-r matrix formed by the top r rows of the input–output matrix $\boldsymbol{B} \in \mathbb{R}^{n \times r}$ is the identity and the rest of the $\boldsymbol{B}$ matrix is zero. Moving into the Laplace domain, (11) led to an expression for the r-by-r multiport impedance matrix for the net complex

$$\boldsymbol{v}(s) = \boldsymbol{Z}(s)\boldsymbol{i}(s) \tag{12}$$

$$\boldsymbol{Z}(s) = \boldsymbol{B}^T(\boldsymbol{G} + s\boldsymbol{C})^{-1}\boldsymbol{B}. \tag{13}$$

We choose impedance macromodeling over admittance macromodeling [25]–[27] because of the ease with which we can fold linearized driver and receiver models into the analysis.

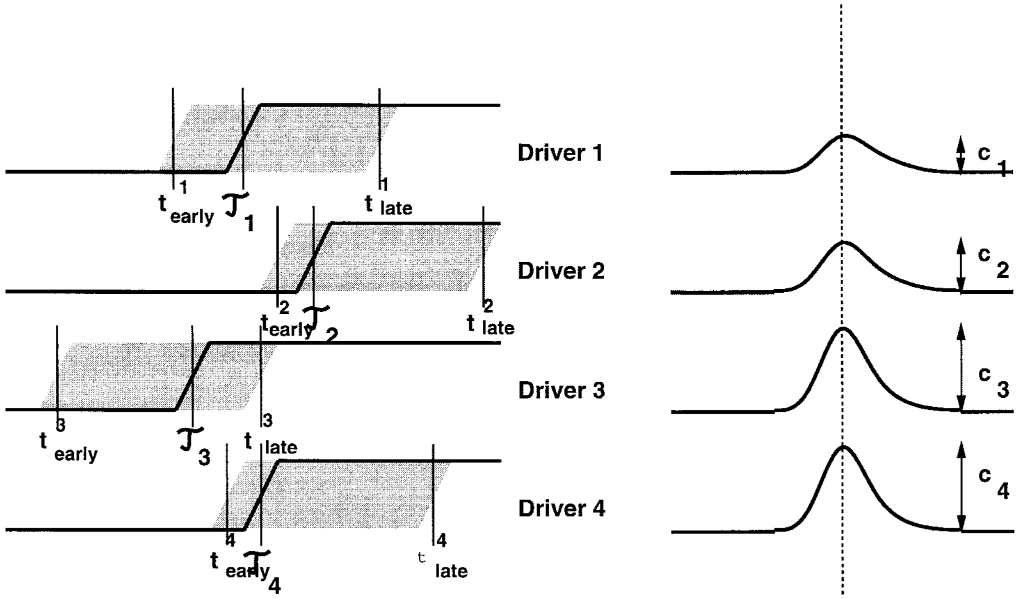


Fig. 15. Timing orthogonality. The switching times τ_i are chosen so that the peak noises align.

Because a net complex in general does not have a dc path to ground, the impedance matrix is singular at $s = 0$. To avoid this singularity, we choose a nonzero expansion point s_o for the moment matching associated with a typical net Elmore delay. Using the change of variable $s = s_o + \tilde{s}$, (13) becomes

$$\boldsymbol{Z}(s_o + \tilde{s}) = \boldsymbol{B}^T(\boldsymbol{\mathcal{G}} + \tilde{s}\boldsymbol{C})^{-1}\boldsymbol{B} \tag{14}$$

where $\boldsymbol{\mathcal{G}} = \boldsymbol{G} + s_o\boldsymbol{C}$. $\boldsymbol{\mathcal{G}}$ will be symmetric positive definite for a choice of real positive s_o. We then employ a multiport symmetric Lanczos process [28], [29] described in detail in [30] which is applicable to symmetric, positive-definite $\boldsymbol{\mathcal{G}}$.

This results in a reduced-order model of order p

$$\boldsymbol{Z}_p(s_o + \tilde{s}) = \tilde{\boldsymbol{B}}_p^T\left(\boldsymbol{I} + \tilde{s}\tilde{\boldsymbol{H}}_p\right)^{-1}\tilde{\boldsymbol{B}}_p \tag{15}$$

where $\boldsymbol{Z}_p \in \mathbb{R}^{r\times r}$ and $p = p_1 + p_2 + \cdots + p_q$ where p_j is the rank of $\boldsymbol{V}_j$. $\tilde{\boldsymbol{H}}_p \in \mathbb{R}^{p\times p}$ is a block tridiagonal matrix such that

$$\tilde{\boldsymbol{H}}_p = \tilde{\boldsymbol{V}}_p^T\boldsymbol{\mathcal{G}}^{-1}\boldsymbol{C}\boldsymbol{\mathcal{G}}^{-1}\tilde{\boldsymbol{V}}_p. \tag{16}$$

It can be further shown that this model is also passive [28]. These interconnect macromodels are stored as DCL binary dynamic tables (BDT) which are subsequently utilized by a DCL interconnect subrule for noise analysis. We take advantage of the sparsity in storing the $\tilde{\boldsymbol{H}}_p$ and $\tilde{\boldsymbol{B}}_p$ matrices. The $\tilde{\boldsymbol{H}}_p$ matrix is symmetric and block tridiagonal. The $\tilde{\boldsymbol{B}}_p$ matrix is zero except for the top r-by-r which is upper triangular.

The noise abstracts generated from the Macro Harmony run are used along with the interconnect macromodels to check the noise on the global interconnect. A DCL interconnect subrule performs the noise calculation from the macromodels loaded with the BDT. We first fold the driver resistance and receiver capacitances from the abstract port modeling for the primary net into the multiport impedance as shown in Fig. 14(b). Following Assumption 7, secondary net drivers are modeled as ideal switching voltage sources. Secondary net receivers are modeled with the associated noise-output port capacitance. The conductance of the primary net driver is obtained from the R_L and R_H values in the noise abstract. Reference [30] describes the details of how the $\boldsymbol{Z}_p$ macromodel is combined with this information to perform the coupled noise calculation.

The Global Harmony architecture shown in Fig. 13 includes a tight coupling with the static timing analysis of the same design. This enables timing information to be used in the calculation of noise. We obtain secondary net driver slews from the timer model. Timing windows, as defined by the earliest and latest possible arrival time, are determined for each secondary net driver. By Assumption 3 of Section V, this allows us to calculate the worst possible noise in the presence of arrival time constraints, reducing pessimism in the analysis. The problem can be formally stated as follows (see Fig. 15). Let c_i be the peak noise on a given primary receiver associated with driver i. Let t^i_{early} be the earliest arrival time associated with secondary driver i and let t^i_{late} be the latest arrival time associated with secondary driver i. In addition, let τ_i be the switching time associated with secondary net driver i, such that all the noise peaks align for the primary receiver in question. Let x_i be the binary variable indicating whether the given secondary net driver is switching, and let n be the number of secondary nets. The problem is then to maximize

$$\sum_{i=1}^{n} c_i x_i \tag{17}$$

such that the following $2n$ constraints can be satisfied for all i:

$$\left(t^i_{\text{late}} - \tau_i - t_{ref}\right)x_i \geq 0 \tag{18}$$

$$\left(\tau_i + t_{ref} - t^i_{\text{early}}\right)x_i \geq 0 \tag{19}$$

where t_{ref} is a continuous variable determining the absolute time reference for the τ_i. In this form, the problem takes the form of a mixed integer programming problem. Alternately, the constraints can be reformulated to remove t_{ref} and consider only relative times. For all $i \neq j$

$$\left(t^i_{\text{late}} - t^j_{\text{early}} - \tau_i + \tau_j\right)x_i x_j \leq 0 \tag{20}$$

$$\left(t^j_{\text{late}} - t^i_{\text{early}} + \tau_i - \tau_j\right) x_i x_j \le 0. \quad (21)$$

We refer to these constraints as *timing orthogonality*. Because t^i_{early} and t^i_{late} result from early and late path propagation in static timing analysis, the timing windows incorporate the switching of the secondary-net drivers due to hazards.

This formulation assumes a certain "sharpness" to the noise peaks. When the peak falls outside the arrival-time window, its contribution is taken as zero. We utilize a branch-and-bound algorithm [31], [32] to solve this problem since the noise on each subtree can be easily bounded by the assumption that each node in that subtree is contributing. The maximum noise of (17) is added to the propagate noise from the noise abstracts for each receiver and compared against the noise margins also contained in the noise abstracts. A global noise "slack" report results. These slacks are based on pessimistic dc noise margins at the macro inputs. As indicated previously, we can eliminate this pessimism by performing a Macro Harmony run on the macro using assertions of the actual input noise generated from Global Harmony.

VI. Results

We begin with a simple, through comprehensive, example that includes pass transistors, static gates, and domino gates to illustrate how static noise analysis acts to pinpoint a functional failure. We consider the results that follow from a rigorous analysis based on the assumptions of Section IV as well as from an analysis based on the additional assumptions employed in Harmony. Consider again the circuit of Fig. 9, in which a domino gate drives a latch. The dynamic node E of the domino gate is capacitively coupled to another switching net. In Fig. 16, we show the dynamics of a functional fail that results from noise for this example. As shown in Fig. 16(a), A is switching from low to high, injecting charge-sharing noise onto node E. Node J' is also switching from high to low, adding coupled noise. At the same time, noise appears on nodes C and D as might occur, for example, as a result of coupling to these nodes [as shown in Fig. 16(b)]. Power-supply noise causes the rails to expand during the evaluate phase of the domino gate. This is typical behavior, since the voltage rails often collapse transiently during precharge. These noise sources *together* are enough to switch the output of the dynamic gate, F, and change the state of the latch (nodes G, H), as shown in Fig. 16(c). CLK is assumed to be high. Since the latch should have a logic one as its output, but instead has a logic zero, functional failure of this hardware will result.

The noise failure demonstrated in Fig. 16 is critically dependent on the contributions of all of the noise sources at work: power-supply noise, charge-sharing noise, coupling noise, and propagated noise. Fig. 17 shows how this noise fail would not occur in the absence of any of these noise sources. Fig. 17(a) shows the node voltages E, G, and H in the absence of input noise on either node C or node D. In Fig. 17(b), we show the voltages in the case that there is no coupling noise; that is, node J' does not switch. In Fig. 17(c), we show the voltages in the case that node A does not switch; that is, there is no charge-sharing noise. We show the voltages in the case that there is no power supply noise in Fig. 17(d).

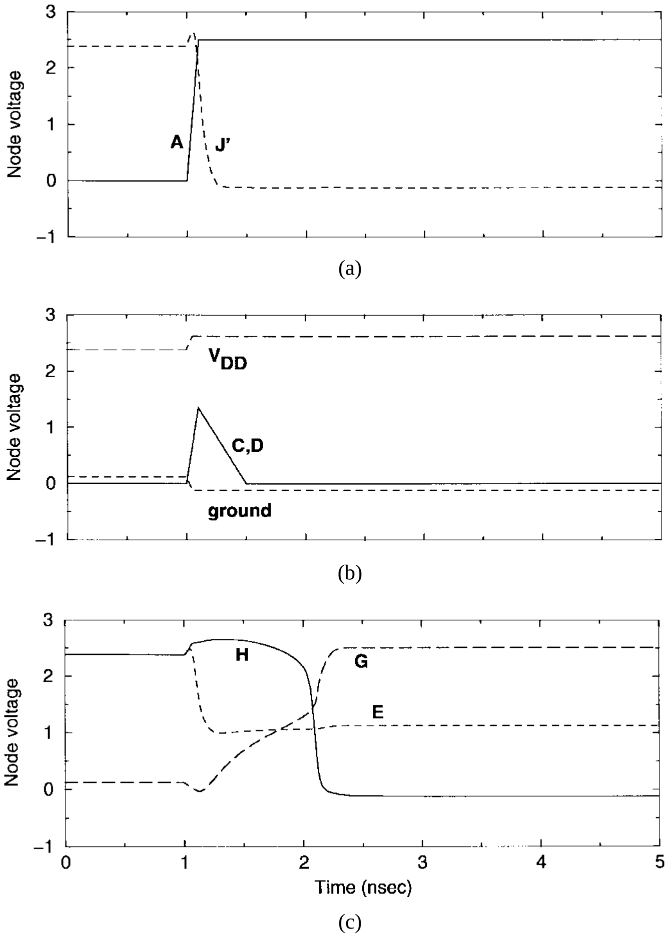
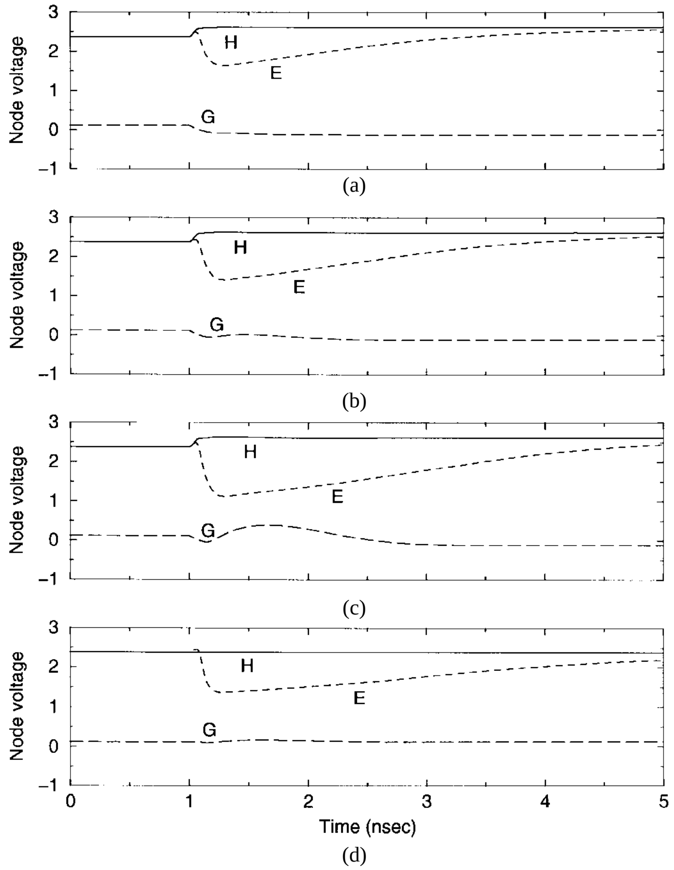


Fig. 16. Circuit simulation of a functional failure due to noise: (a) driver output net J' is switching as is node A, (b) coupling noise appears on inputs C and D along with power-supply noise on the voltage rails, and (c) the dynamic node (E) falls, switching the output inverter of the domino gate (F) and the latch output (G).

Even though this noise failure results from a complex interaction of several noise sources, static noise analysis of this network precisely predicts the problem. We perform this analysis in two ways: using the noise graph of Fig. 10(a) which applies only the assumptions of Section IV; and using the noise graph of Fig. 10(b), applying the additional assumptions employed in Harmony. In both analyses, we must first calculate the worst-possible V_H noise which can appear on node E (Assumption 4). This worst case sensitization involves superposition of the charge-sharing noise injected by the switching of node A, the noise injected from the noise-input port C, the noise injected from the noise-input port D, and the interconnect coupling noise injected by the switching of node J'. (We actually would have to first check the stability of the domino gate to the noise sources at C and D before performing this analysis. In this example, the domino gate is, in fact, stable in the presence of this noise.)

To calculate each of these noise components, we establish the network shown in Fig. 18. Gate inputs are treated as linear capacitors tied to ground (Assumption 1). The driver at J' is replaced by a independent voltage source (Assumption 7). The sensitization producing the worst total V_H noise response at node E has C, D, and F all set to zero and B set to one.



(a)

(b)

(c)

(d)

Fig. 17. Noise failure will not occur in the absence of (a) injected noise on either input C or input D, (b) coupling noise, (c) charge-sharing noise, or (d) power-supply noise.

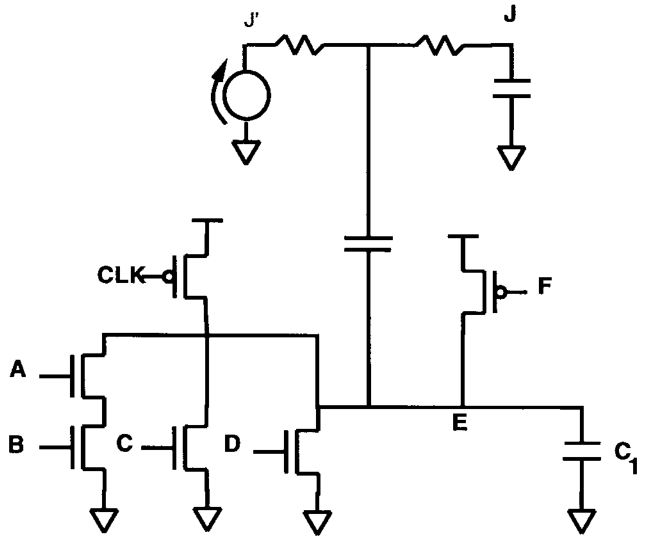
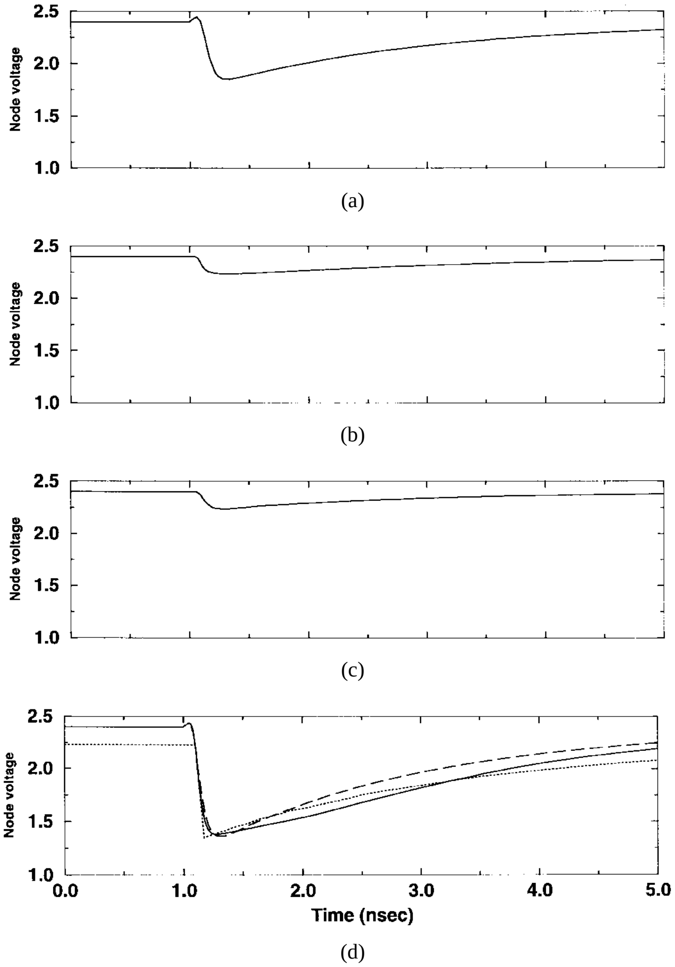


Fig. 18. Network for the simulations to compute the noise at node E.

V_L noise is appearing on C and D. J' is switching from high to low, and A is switching from low to high. CLK is one for the evaluate phase of this dynamic gate. The capacitor C_1 represents the gate capacitance of transistors $M2$ and $M3$ in Fig. 9. We calculate each noise source acting independently, using $V_{DD}^{\rm min}$ for supply and Gnd$^{\rm max}$ for ground to account for the power-supply fluctuations. In Fig. 19(a), we show the coupling noise appearing on node E due to the switching of J'. In Fig. 19(b), we show the charge-sharing noise calculated on node E due to the switching of node A. Fig. 19(c) shows the noise propagated to node E due to V_L noise injected onto node C. The exact same curve results for V_L noise propagated to node E from node D under comparable sensitization conditions. In Fig. 19(d), we show the noise that results by superposing in the time-domain the results of graphs (a)–(c) with (c) contributing twice. Special consideration is made not to double count base level in this superposition. All of the peak noises are aligned (Assumption 3). We compare this result with a full simulation, shown as the dashed curve in Fig. 19(d) with excellent agreement. In the Harmony implementation, the propagated noise is treated as dc, and the voltage waveform at E is abstracted as the dotted curve in Fig. 19(d).



(a)

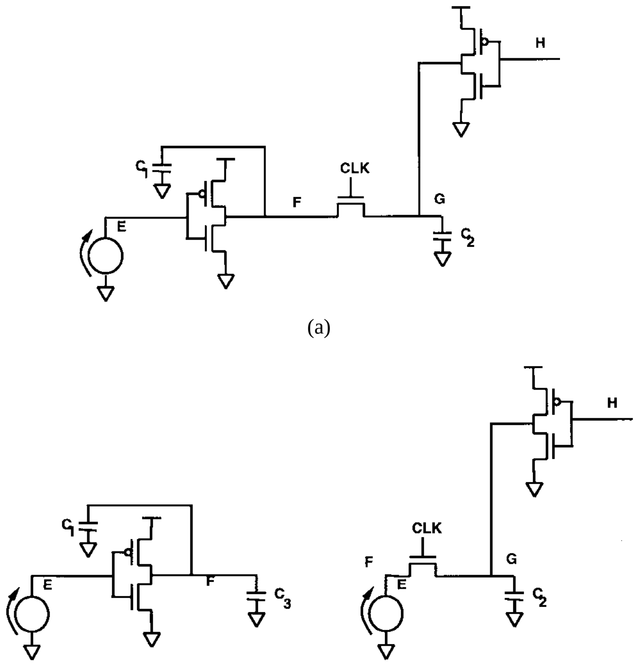
(b)

(c)

(d)

Fig. 19. Noise calculation at node E: (a) coupled noise due to switching driver at J', (b) noise due to charge-sharing from switching input A, (c) noise propagated from node C or D, and (d) superposed noise. The solid curve in (d) comes from a strict time-domain sum. The dashed curve shows the exact result from circuit simulation.

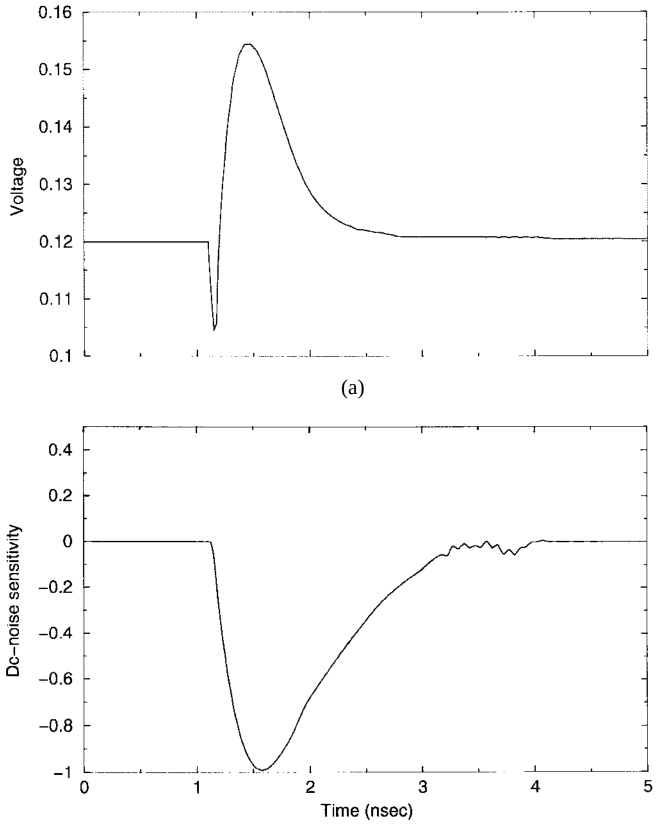
Having calculated the total noise appear on node E and following the noise graph, we must now propagate this noise across the next stage and perform the associated noise stability check which should flag a possible violation. In this case the analyses associated with Fig. 10(a) and (b) differ. Fig. 20(a) shows the network for this analysis associated with the graph of Fig. 10(a), while Fig. 20(b) shows the two-stage calculation



(a)

(b)

Fig. 20. Networks to perform the stability check and propagate the noise from node E to node G: (a) for the noise graph of Fig. 9(a) and (b) for the noise graph of Fig. 9(b).



(a)

(b)

Fig. 21. Noise calculation at node G for the network of Fig. 19(a): (a) propagated noise at node G and (b) time-domain dc-noise sensitivity of the output noise at this input.

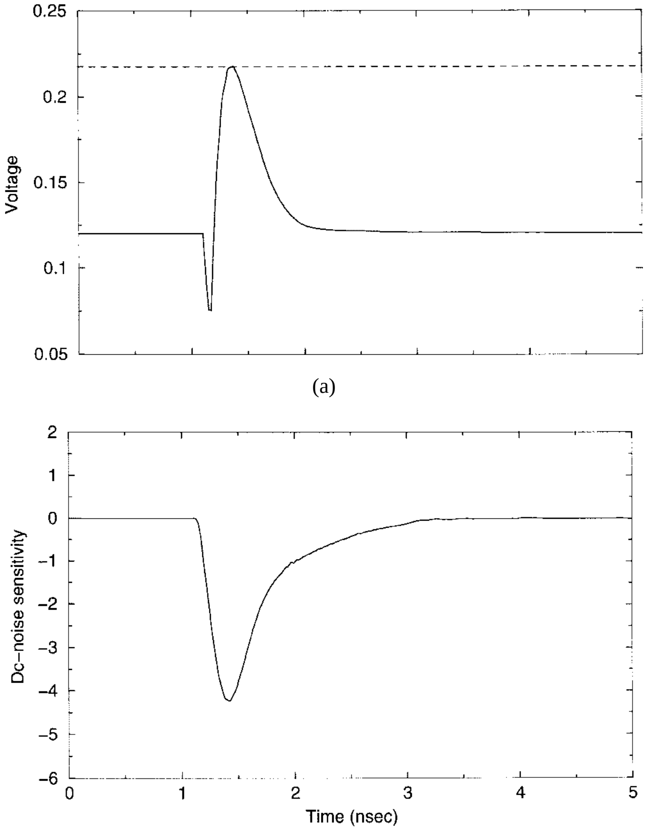
corresponding to the noise graph of Fig. 10(b) used in Harmony. We first consider the analysis of Fig. 20(a). Because we are still in the evaluate phase, CLK is one. The capacitor C_1 represents the gate capacitance of transistor $M1$ in Fig. 9, while C_2 represents the gate capacitance of transistors $M6$ and $M7$. In Fig. 21(a), the noise propagated to node G based on this analysis is shown, while Fig. 21(b) displays the time-domain sensitivity of this noise to dc variations on node E. The peak sensitivity is almost exactly -1, indicating that the inverter is biased to the verge of a noise instability. The Harmony implementation, based on an analysis of Fig. 20(b), begins with the calculation of the noise appearing on node F using the noise stimulus defined by the dotted curve in Fig. 19(d). Because all propagated noise is treated as dc, the dashed curve shows the equivalent dc level propagated in Harmony. Fig. 22(b) shows the corresponding dc-noise time-domain sensitivity for the path from E to F. A stability violation is also reported with the magnitude of the peak sensitivity greater than it is in Fig. 21(b). This additional pessimism is associated with the reduced loading assumption which comes with assuming the pass gate is off in calculating the capacitance C_3.

In this example, static noise analysis has located a potential functional failure due to noise. A natural to question to ask in general is: how do we know the failure is not false? There are two main sources of pessimism in static noise analysis. The first is the conservatism of the metric; that is, a circuit may still be functional even though static noise analysis indicates a noise instability. For example, in the circuit of Fig. 9, if the $M1$ half-latch device is removed, the latch will not falsely change state even though the $M2$–$M3$ inverter will still be driven unstable. Allowing noise instability in such cases is generally unwise. Because a restoring logic gate is biased into its high-gain region of operation, the circuit is very sensitive to the slightest process variation or error in the analysis. Noise instabilities are serious design weaknesses which should be corrected almost without exception. The second major source of pessimism in static noise analysis is the worst case temporal correlation assumption combined with the possibility that the sensitization may not be logically possible. Timing constraints and logic constraints help significantly in most cases to reduce this pessimism. Even if a noise instability is found that is consistent with *possible* timing and logical relationships, this situation may "never" occur in normal machine operation. The problem in this case is that "never" is extremely difficult to quantify. As a result, sage design practice requires that any *possible* noise failure be eliminated from the design, however remote the possibility. One must be able to safely rule-out noise as a failure mechanism in manufacturing test.

Macro Harmony has been run all of the macros of a high-frequency S/390 microprocessor design. In the final runs, all designs were free of noise stability violations. Table III shows some run-time results for several representative macros. All runs were done on an RS/6000, Model 590. The table also shows the number of transistors, the number of evaluation nodes in the noise graph, the number of channel-connected components, the number of gates, and the number of pass gates. On the average, 40%–45% of the run time is spent in the circuit simulation engine. The run time is dependent on not only the size of the macro but the circuit topology.

TABLE III
RESULTS FOR MACRO HARMONY RUNS ON A REPRESENTATIVE SET OF MACROS FOR A HIGH-FREQUENCY S/390 DESIGN

circuit name	type	# transistors	# CCCs	# nodes	# gates	# pass gates	CPU time
epdalm	static	160	41	62	41	0	1m 48s (108 s)
idud	static	180	48	74	48	0	2m 4s (124 s)
roe	dynamic	1055	173	383	553	398	1m 5s (65 s)
idia	static	1186	328	429	420	81	2m 59s (179 s)
bal	static	8645	2421	2502	2914	482	17m 2s (1022 s)
brqa	static	9616	2317	2428	3472	1025	5m 27s (327 s)
rwspi	static	13436	3728	3796	3692	653	19m 42s (1182 s)
btadd	dynamic	17280	4205	4511	5913	1743	21m 51s (1311 s)
brqb	static	23340	5866	6027	8492	1990	21m 19s (1279 s)
btp	dynamic	25708	6991	7224	10002	2514	49m 53s (2993 s)
ro_super	static	36011	10902	11412	11886	2717	61m 22s (3682 s)
ria_super	static	80665	22852	23544	31175	7655	154m 31s (9271 s)
epdmpy	static	120683	27788	28373	52240	33295	197m 43s (11863 s)



(a)

(b)

Fig. 22. Noise calculation at node F for the network of Fig. 19(b): (a) propagated noise at node F and (b) time-domain dc-noise sensitivity of the output noise at this input.

Circuits containing pass gates run slower because of the more complex preconditioning required. This implementation relies on the ACES [19] timing simulator as the simulation engine.

In Table IV we show typical "noise slack" results for a Global Harmony run on fixed-point unit of a high-performance CMOS S/390 microprocessor design. This section of the chip has 4031 receivers. The noise tolerance at each input is set to zero for this run so that the full spectrum of the coupling noise can be observed. The results are shown with and without timing orthogonality to remove pessimism. Timing orthogonality is most effective in eliminating the number of high-noise "outliers." The supply voltage is 1.8 V.

TABLE IV
NOISE SLACK RESULTS FOR A GLOBAL HARMONY RUN ON A SECTION OF A HIGH-PERFORMANCE S/390 MICROPROCESSOR WITH 4031 RECEIVERS. THE SUPPLY VOLTAGE IS 1.8 V. RESULTS ARE SHOWN WITH AND WITHOUT TIMING ORTHOGONALITY CONSIDERED

Noise slack range	Number of receivers (aligned)	Number of receivers (orthogonal)
-1 < x < -0.95	1	0
-0.95 < x < -0.9	1	0
-0.9 < x < -0.85	4	4
-0.85 < x < -0.8	9	9
-0.8 < x < -0.75	10	9
-0.75 < x < -0.7	17	14
-0.7 < x < -0.65	126	24
-0.65 < x < -0.6	76	59
-0.6 < x < -0.55	101	96
-0.55 < x < -0.5	363	358
-0.5 < x < -0.45	481	361
-0.45 < x < -0.4	501	634
-0.4 < x < -0.35	716	596
-0.35 < x < -0.3	684	670
-0.3 < x < -0.25	1012	1112
-0.25 < x < -0.2	1119	1068
-0.2 < x < -0.15	940	1115
-0.15 < x < -0.1	1883	1910
-0.1 < x < -0.05	12	7
-0.05 < x < 0.0	6	12

VII. CONCLUSIONS

In this paper, we have defined noise and discussed the noise sources relevant to digital systems. We have also defined a metric, noise stability, for providing a formal basis for verifying the noise immunity of a digital integrated circuit. We have then described the techniques of static noise analysis within the context of Harmony, a two-level hierarchical implementation used in the design of complex, high-frequency microprocessors. Macro Harmony combines transistor-level analysis with circuit simulation. Global Harmony incorporates a unique combination of timing and noise analysis and employs a reduced-order modeling algorithm that allows for passive interconnect macromodeling and efficient storage of the macromodel result.

ACKNOWLEDGMENT

The authors would like to acknowledge the significant contributions of P. C. Elmendorf and G. Zheng to the development of Global Harmony. The authors also acknowledge many useful discussions with A. Elfadel, J. Beatty, A. Suess, E. Chiprout, A. Gupta, C. Visweswariah, J. Rahmeh, and P. Villarrubia. The successful application of the Harmony implementation of static noise analysis within IBM would not have been possible without the contributions of the microprocessor design teams in Poughkeepsie and Yorktown.

REFERENCES

[1] P. R. Gray and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits.* New York: Wiley, 1984.

[2] D. K. Su, M. J. Loinaz, S. Masui, and B. A. Wooley, “Experimental results and modeling techniques for substrate noise in mixed-signal integrated circuits,” *IEEE Trans. Computer-Aided Design*, vol. 28, pp. 420–430, Apr. 1993.

[3] J. A. Olmstead and S. Vulih, “Noise problems in mixed analog-digital integrated circuits,” in *Proc. Int. Custom Integrated Circuits Conf.*, 1987, pp. 659–662.

[4] K. L. Shepard and V. Narayanan, “Conquering noise in deep submicron digital design,” *IEEE Design Test Comput.*, pp. 51–62, Jan./Mar. 1998.

[5] R. B. Hitchcock, G. L. Smith, and D. D. Cheng, “Timing analysis for computer hardware,” *IBM J. Res. Dev.*, vol. 26, pp. 100–105, 1982.

[6] K. L. Shepard, S. Carey, E. Cho, B. Curran, R. Hatch, D. Hoffman, S. McCabe, G. Northrop, and R. Seigler, “Design methodology for the G4 S/390 microprocessors,” *IBM J. Res. Dev.*, vol. 21, nos. 4/5, pp. 515–548, 1997.

[7] A. Vittal and M. Marek-Sadowska, “Crosstalk reduction for VLSI,” *IEEE Trans. Computer-Aided Design*, vol. 16, pp. 290–298, Mar. 1997.

[8] D. A. Kirkpatrick and A. L. Sangiovanni-Vincentelli, “Techniques for crosstalk avoidance in the physical design of high performance digital systems,” in *ICCAD’94*, San Jose, CA, Nov. 1994, pp. 130–133.

[9] ——, “Digital sensitivity: Predicting signal interaction using functional analysis,” in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 1996, pp. 536–541.

[10] L. Miller, “Controlled collapse reflow chip joining,” *IBM J. Res. Dev.*, vol. 13, no. 3, pp. 239–250, 1969.

[11] H. H. Chen, “Minimizing chip-level simultaneous switching noise for high-performance microprocessor design,” in *Proc. IEEE Int. Symp. Circuits and Systems*, 1996, vol. 4, pp. 544–547.

[12] C. F. Hill, “Noise margin and noise immunity of logic circuits,” *Microelectron.*, 1968, vol. 1, pp. 16–21.

[13] J. Lohstroh, “Static and dynamic noise margins of logic circuits,” *IEEE J. Solid-State Circuits*, vol. SC-14, pp. 591–598, June 1979.

[14] J. Lohstroh, E. Seevinck, and J. De Groot, “Worst case static noise margin criteria for logic circuits and their mathematical equivalence,” *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 803–806, Dec. 1983.

[15] E. Seevinck, F. List, and J. Lohstroh, “Static-noise margin analysis of MOS SRAM cells,” *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 748–754, Oct. 1987.

[16] L. A. Glasser and D. W. Dobberpuhl, *The Design and Analysis of VLSI Circuits.* Reading, MA: Addison-Wesley, 1985.

[17] J. M. Zurada, Y. S. Joo, and S. V. Bell, “Dynamic noise margins in MOS logic gates,” in *Proc. Int. Conf. Circuits and Systems*, 1989, pp. 1153–1156.

[18] G. A. Katopis, “Delta-I noise specification of a high-performance computing machine,” *Proc. IEEE*, vol. 73, pp. 1405–1415, 1985.

[19] A. Devgan and R. A. Rohrer, “Adaptively controlled explicit simulation,” *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 746–761, June 1994.

[20] A. Kuehlmann, A. Srinivasan, and D. P. Lapotin, “Verity—A formal verification program for custom CMOS circuits,” *IBM J. Res. Dev.*, vol. 39, nos. 1/2, pp. 149–165, 1995.

[21] N. P. Jouppi, “Derivation of signal flow direction in MOS VLSI,” *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 480–490, 1987.

[22] K.-J. Lee, C.-N. Wang, R. Gupta, and M. A. Breuer, “An integrated system for assigning signal flow directions to CMOS transistors,” *IEEE Trans. Computer-Aided Design*, vol. 14, no. 12, pp. 1445–1458, Dec. 1995.

[23] R. E. Bryant, “Graph-based algorithms for Boolean function manipulation,” *IEEE Trans. Computer-Aided Design*, vol. CAD-35, pp. 677–691, 1986.

[24] “Delay calculation language (DCL) and procedural interface (PI),” Tech. Rep., CAD Framework Initiative, Version 1.0.1.

[25] V. Raghavan, J. Eric Bracken, and R. A. Rohrer, “AWESpice: A general tool for the accurate and efficient simulation of interconnect problems,” in *Proc. 29th ACM/IEEE Design Automation Conf.*, Anaheim, CA, June 1992, pp. 87–92.

[26] S. Y. Kim, N. Gopal, and L. T. Pillage, “AWE macromodels of VLSI interconnect for circuit simulation,” in *Proc. IEEE/ACM Int. Conf. Computer Aided-Design*, Santa Clara, CA, Nov. 1992, pp. 64–70.

[27] ——, “Time-domain macromodels for VLSI interconnect analysis,” *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 1257–1270, 1994.

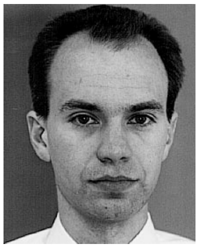
[28] R. W. Freund and P. Feldmann, “Reduced-order modeling of large passive linear circuits by means of the SyPVL algorithm,” in *Proc. ICCAD’96*, San Jose, CA, Nov. 1996, pp. 280–287.

[29] R. W. Freund and N. M. Nachtigal, “Software for simplified Lanczos and QMR algorithms,” *Appl. Numer. Math.*, vol. 19, pp. 319–341, 1995.

[30] K. L. Shepard, V. Narayanan, P. C. Elmendorf, and G. Zheng, “Global harmony: Coupled noise analysis for full-chip RC interconnect networks,” presented at *IEEE Int. Conf. Computer-Aided Design*, 1997.

[31] G. Nernhauser and L. Wolsey, *Integer and Combinatorial Optimization.* New York: Wiley, 1988.

[32] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization.* Englewood Cliffs, NJ: Prentice-Hall, 1982.



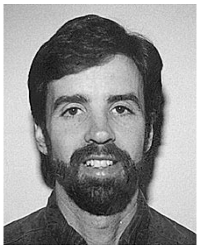
Kenneth L. Shepard (S’85–M’91) received the B.S.E. degree from Princeton University, Princeton, NJ, in 1987 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1988 and 1992, respectively.

From 1992 to 1997, he was a Research Staff Member and Manager in the VLSI Design Department at the IBM T. J. Watson Research Center. Since 1997, he has been an Assistant Professor of Electrical Engineering at Columbia University as well as Chief Technology Officer of CadMOS Design Technology.

At IBM, Dr. Shepard worked on the design of the G4 S/390 microprocessor for which he received Research Division Awards in 1995 and 1996. He was the recipient of an NSF CAREER Award and IBM Early Faculty Development Award in 1998 and the 1999 Distinguished Faculty Teaching Award from the Columbia Engineering School Alumni Association. He is an Associate Editor of IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS.

Vinod Narayanan (S’88–M’89–SM’96) received the M.S. and Ph.D. degrees from Syracuse University, Syracuse, NY, in 1987 and 1989, respectively.

He is currently a Vice President of Research and Development at CadMOS Design Technology. Prior to joining CadMOS, he was a Research Staff Member at IBM T. J. Watson Research Center, working on various design automation problems. He is interested in many areas of design automation, including physical design, floorplanning, timing analysis, signal integrity analysis, and DA for highly integrated systems. His other interests include computer security and privacy, cryptography, object oriented design, and software architecture.



Ron Rose received the B.S. degree in electrical engineering in 1982 from the University of Notre Dame, Notre Dame, IN.

He is a Senior Engineer in IBM’s Electronic Design Automation Project, which he joined in 1982 to work on simulator development. Since 1996, he has been working primarily on the development of noise analysis tools.