Generating Realistic Stimuli for Accurate Power Grid Analysis

P. MARQUES MORGADO and PAULO F. FLORES
INESC ID/IST - TU Lisbon
and
L. MIGUEL SILVEIRA
INESC ID/Cadence Research Laboratories/IST - TU Lisbon

Power analysis tools are an integral component of any current power sign-off methodology. The performance of a design's power grid affects the timing and functionality of a circuit, directly impacting the overall performance. Ensuring power grid robustness implies taking into account, among others, static and dynamic effects of voltage drop, ground bounce, and electromigration. This type of verification is usually done by simulation, targeting a worst-case scenario where devices, switching almost simultaneously, could impose stern current demands on the power grid. While determination of the exact worst-case switching conditions from the grid perspective is usually not practical, the choice of simulation stimuli has a critical effect on the results of the analysis. Targetting safe but unrealistic settings could lead to pessimistic results and costly overdesigns in terms of die area. In this article we describe a software tool that generates a reasonable, realistic, set of stimuli for simulation. The approach proposed accounts for timing and spatial restrictions that arise from the circuit's netlist and placement and generates an approximation to the worst-case condition. The resulting stimuli indicate that only a fraction of the gates change in any given timing window, leading to a more robust verification methodology, especially in the dynamic case. Generating such stimuli is akin to performing a standard static timing analysis, so the tool fits well within conventional design frameworks. Furthermore, the tool can be used for hotspot detection in early design stages.

Categories and Subject Descriptors: J.6. [**Computer Applications**]: Computer-Aided Engineering—*Computer-aided design (CAD)*; I.6.4 [**Simulation and Modeling**]: Model Validation and Analysis

General Terms: Algorithms, Verification

This research was supported in part by the Portuguese FCT under program POSI, ref. EEA-ESE/61528/2004.

Authors' addresses: P. M. Morgado, INESC ID, Department of Electrical and Computer Engineering, IST - Technical University of Lisbon, Rua Alves Redol, 9, 1000-029 Lisboa, Portugal; email: morgado@algos.inesc-id.pt; P. F. Flores and L. M. Silveira, INESC ID, Department of Electrical and Computer Engineering, IST - Technical University of Lisbon, Rua Alves Redol, 9, 1000-029 Lisboa, Portugal; email: {pff,lms}@inesc-id.pt.

DOI 10.1145/1529255.1529262 http://doi.acm.org/10.1145/1529255.1529262

Additional Key Words and Phrases: Power grid, verification, simulation, voltage drop, ground bounce, stimuli generation

ACM Reference Format:

Morgado, P. M., Flores, P. F., and Silveira, L. M. 2009. Generating realistic stimuli for accurate power grid analysis. ACM Trans. Des. Autom. Electron. Syst. 14, 3, Article 40 (May 2009), 26 pages, DOI = 10.1145/1529255.1529262 http://doi.acm.org/10.1145/1529255.1529262

1. INTRODUCTION

Power distribution system design is an area of increasing concern in the semiconductor industry. According to data from a Cadence whitepaper [2001], more than 50% of a large number of tapeouts using 0.13-micron technology would fail if the power distribution system were not validated beforehand. Lower operating voltages, increased device integration density and leakage currents, higher operating frequencies, and the use of low-power design techniques all tend to stress design's power grids as technology evolves. The design of such systems is complex and error prone, since there is a wide variety of aspects that must be taken into account. Of these, perhaps the four major problems that may affect power distribution systems are voltage drop, ground bounce, L di/dt noise, and electromigration [Lin and Chang 2001].

Voltage drop, also called IR drop, is the voltage reduction that occurs on power supply networks. The IR drop can be static or dynamic and results from the existence of nonideal elements: the resistance within the power and ground supply wiring and the capacitance between them. While static voltage drop considers only the average currents, dynamic voltage drop considers current waveforms within clock cycles and has an RC transient behavior. Similar effects may be found in ground wiring, usually referred as ground bounce, whereby current flows back to the ground/VSS pins causing its voltage to fluctuate. Both effects contribute to lower operating voltages within devices (i.e., logic cells/gates in digital circuits), which in general increases the overall time response of a device and might cause operational failures. L di/dt noise is caused by current spikes on wires that will induce abrupt voltage changes on these wires and their neighboring wires, due to inductance coupling [Sato et al. 2000; Choi et al. 2002]. Finally, electromigration in the power distribution system is the movement of metal atoms from a certain region to another, and is caused by high current densities. Electromigration can itself lead to voltage drop and/or ground bounce as metal lines wear out.

Several approaches can be taken to minimize or eliminate these problems such as the insertion of decoupling capacitance (*decap*) and/or the use of wider metal lines. Decoupling capacitance work as a "charge reservoir," in a situation where several devices become active and the power grid is unable to deliver the total required current. The location of the decoupling capacitors within the power grid is extremely important. Badly placed *decaps* may have a reduced contribution for voltage drop and ground bounce elimination and are a waste of die area, since in general they consume a considerable amount of silicon [Su et al. 2003].

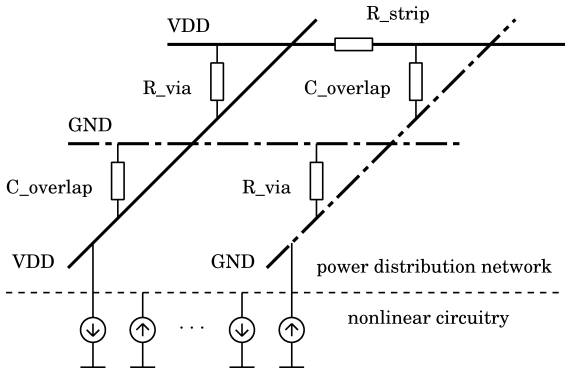

Fig. 1. Simplified power grid model.

Power grid analysis and verification is, from a practical standpoint, one of the most important steps in the design flow, yet computationally a very complex one. Power grid verification is usually accomplished by simulation [Nassif and Kozhaya 2000; Zhong and Wong 2005]. The disadvantage of simulation is that stimuli must be generated very carefully such that the relevant scenarios are accounted for. Only settings corresponding to the chosen stimuli are simulated and thus verified, so they should be chosen appropriately and should be representative of relevant situations. Since the power grid encompasses the whole die area, its description is rather large and the simulation process is slow and highly complex. This results from the necessity to take into account a huge number of power grid parameters (RLC nonidealities) and all the devices that take current from it, as shown in Figure 1.

Electrically simulating the power grid with all the devices is not practical for today's designs. As such, most power analysis tools in use today model devices (transistors) with simplified networks, sometimes only consisting of current sources. Such sources supposedly emulate the switching activity of the circuit, thus enabling grid verification. Generating realistic descriptions for the waveforms at these sources is key to efficient and robust grid analysis. Given the size and complexity of circuits, automatic ways of generating realistic sets of stimuli are necessary. Hopefully, this task can be integrated in a standard design and verification framework and accomplished efficiently in an acceptable time. In fact, most power sign-off methodologies available today from the leading EDA vendors include options for vector and vectorless analysis. In vector-based analysis, the designer provides the stimuli for simulation. In vectorless mode, the tool automatically generates such stimuli based on circuit connectivity, functionality, and possibly estimates of switching activity. Once the stimuli are generated, analysis proceeds normally.

In this article we describe an algorithm for generating a reasonable realistic set of stimuli for simulation. The algorithm has been implemented in a software tool that has been integrated into the Cadence VoltageStorm toolset for development and comparisons. The approach pursued takes into account the timing functionality of the design, as well as spatial information available from the circuit's netlist and placement. Voltage drop and ground bounce may occur if

there is a significant number of devices becoming active in a short period of time and drawing current from close regions of the power grid. Combining timing and spatial constraints allows us to determine approximately, within a given time window, how many devices may become active on nearby regions of the power grid. The higher the number of devices in this situation, the greater the possibility that voltage drop and/or ground bounce effects adversely affect the power grid's behavior. The tool is highly configurable in terms of the timing and spatial views of the design and can be used to generate a signature of the circuit's switching behavior, from which current waveforms can be derived using available knowledge from the individual cell's behavior. Generating such stimuli is akin to performing a standard static timing analysis, so the tool is very efficient and fits well within conventional design frameworks. Furthermore, the tool can also be used for hotspot detection in early design stages.

The outline of this article is as follows: In Section 2, we review existing techniques for ensuring grid robustness and present the traditional design flow with emphasis on the power grid design, analysis, and generation of stimuli. In Section 3 a new design flow is proposed in order to determine a set of more realistic power grid stimuli. Such flow combines placement information with timing data, as presented in Section 4, in order to zoom in on troublesome areas and detect realistic worst-case settings. A more detailed explanation of our method and a description of a tool developed to test the ideas is presented in Section 5, while additional considerations are discussed in Section 6. Finally the results obtained using the proposed method are presented in Section 7, and the conclusions drawn from these results are presented in Section 8.

2. BACKGROUND

In this section we discuss the relevance of power grid verification as well as how this can be achieved at different stages of the design, and we review several procedures to achive this goal, including commercial options for this task.

2.1 Ensuring Grid Robustness

Power grid verification is of critical importance to ensure reliable performance of a design. Guaranteeing proper bias to all devices in a circuit is a necessity in order to make sure that correct behavior is achieved. If at some point certain devices request current from the power grid in order to perform their function, for instance, to switch logic levels or to charge some capacitor, not only must this current be immediately available, but the power supply fluctuation caused by the current flow in the grid cannot exceed some safety threshold. If either of these conditions is not met, it is likely the circuit will not behave as expected or may even malfunction entirely.

Given the complexity of this verification task, it is usually performed in various stages. For instance, early power grid verification can be accomplished even when precise knowledge of the circuit behavior is not readily available. Such a verification can be accomplished prior to placement and can in fact help guide placement as an additional constraint to the process. Clearly in this situation only incomplete information about the grid requirements is available.

However, in order to guarantee that the resulting information is relevant, realistic circuit behavior and structure must be taken into account. Given the lack of precise knowledge regarding the circuit behavior and the constraints it will place on the grid, usually such analysis is performed assuming, for instance, bounds on the peak current that will be requested from the power grid. Considerable research has been dedicated to this problem. In Kouroussis and Najm [2003], global constraints on current sources or sub/groups of current sources are imposed, together with additional local constraints on the maximum current drawn. Enforcing these peak current constraints leads to a linear program, whose solution is comprised of the voltages drops whose magnitude can then be checked. While interesting and applicable in early design stages, the verification provided by this procedure is essentially that of a DC problem and the current estimates, although made more realistic given the usage of local as well as global constraints, are really limit values, leading still to quite pessimistic results.

In Kouroussis and Najm [2005] this approach is further extended with the introduction of the concept of localized area of a power grid, essentially allowing designer to hierarchically build and verify the grid. Together with macromodeling, the analysis process can be greatly simplified. Still, the approach followed here is aimed at early power grid verification and makes use of limit assumptions on the input currents. It has the advantage that no detailed current signatures are assumed or needed but it has the disadvantage that it is in essence a static analysis procedure. Nevertheless, the partitioning ideas are quite interesting as they allow to concentrate on specific regions of the design and also enable usage of an iterative refinement procedure. As we shall discuss later on in Section 3, our proposed approach also uses partitioning concepts to simplify the problem of grid verification. However, the partitioning here is different, given that it targets an early design stage, and input from the designer fuels the partitioning scheme in an efficient manner. In Ferzli et al. [2007] similar ideas are extended beyond simple static analysis, to compute bounds on the grid voltages drops assuming dynamic behavior. Computation of such bounds involve choosing a Δt that is small enough to capture the transition times on the grid voltages but large enough so that it does not lead to overly pessimistic results. Static constraints are still used throughout and the end goal is to find a bound on the peak voltage drop.

Generally, the approaches presented in the references mentioned and in other related work represent very relevant contributions to the problem of power grid verification. They describe a set of conditions that can be imposed on a grid and which, when satisfied, guarantee robustness of the grid. This is achieved by setting up a set of equations involving the voltage drop at the grid nodes, assuming an upper bound on the peak values of the excitations to the grid nodes, that is, the currents are drawn from the grid in a global or local sense and then we solve the resulting system. If solutions to the system of equations can be found, then it can be verified that the grid is indeed robust in the sense that, even in the worst case, the voltage drops are guaranteed to not exceed some predetermined maximum. Typically detailed current signatures or dependencies between device behavior can be hard to capture and are not

directly taken into account. For instance, it is hard at this level to realize that considerable correlations exist between grid nodes because of the structure of the circuitry connected to it. Such correlations imply that certain nodes cannot possibly draw the maximum current simultaneously, a fact that is next to impossible to determine without tracing the signal paths in the circuit. As a consequence, the bounds used at this stage have an inherent pessimism associated to them. This is acceptable in early design stages where such details may not be known or may be subject to considerable changes and are thus unreliable anyway. However, this implies that the resulting analysis is conservative and sometimes overly pessimistic.

For sign-off verifications, more detailed information regarding the underlying circuit behavior is readily available and should be taken advantage of. In particular, dynamic information regarding the circuit behavior, which implicitely contains relevant information about circuit-imposed correlations, may lead to bounds that tend to be much looser, leading to less conservative estimates. In essence the two approaches are complementary and should probably be used together. In early design stages, the more conservative bound-based approach can be used to guide the grid design, perhaps accepting minor potential violations which can be rechecked at a later stage, using more detailed information in a simulation-based verification flow. Such potential violations may turn out to be false positives when additional information is brought into the analysis. If not, then the grid may need to be locally corrected. Our goal in this work is to generate a specific set of stimuli that will be used in the later, simulation-based verification steps. Such stimuli should be realistic and should be able to pinpoint, as much as possible, relevant, realistic worst-case operating conditions.

2.2 Traditional Power Grid Sign-Off Verification

As mentioned previously, simulation is the most commonly used method for sign-off validation of the power grid. It enables to verify if the power grid is suited for a given design, that is, if it is robust enough to deal with problems such as voltage drop and ground bounce. In Figure 2 a simplified version of a standard design flow, with emphasis in power grid design and analysis, and targeting sign-off analysis, is presented (in other words, the flow is seen from a power grid-analysis-centric viewpoint, assuming late design stages where detailed place-and-route information is available).

As can be seen in Figure 2 a typical flow starts with a circuit description in VHDL or Verilog. This description is converted into a gate-level netlist of a given technology library during logical synthesis. After synthesis, place and route of circuit cells is done. To ensure the circuit timing sign-off, Static Timing Analysis (STA) is usually done afterwards. If STA fails, a new place and route should be performed. Then, power grid planning is done or improved, based on the knowledge of power distribution along the circuit [Krodel 1991]. However, this knowledge is, of course, rather limited.

After the power grid design, a simulation (at electrical level) of the grid along with the "circuit" is performed. For this purpose, an RC extraction is done as

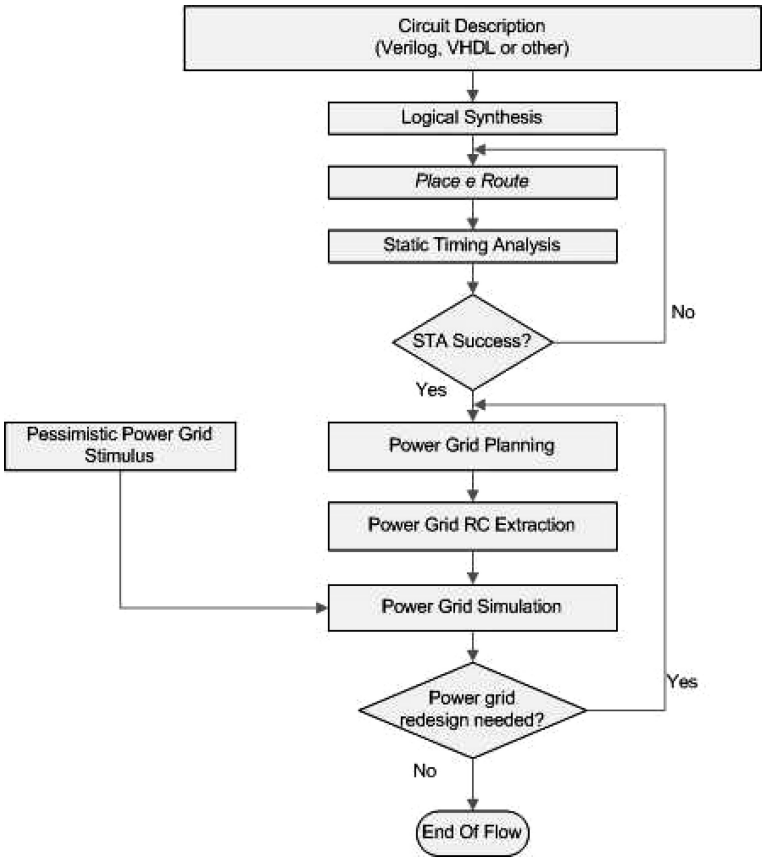

Fig. 2. Traditional design flow (with emphasis on power grid design).

well as the definition of power grid stimuli. This mainly consists of the definition of the circuit cells, which must be considered throughout the simulation, and of their corresponding current waveforms. This or similar methodologies are commonly used and they all share the same problem: the definition of power grid stimuli for an accurate power grid simulation.

This is typically done independently from the flow, although most tools nowadays provide options for automatic, vectorless input generation. Many different methodologies are in use or have been reported for this task. Some of the earlier simulation tools considered all the circuit devices as stimuli to the power grid and assumed as a worst case that all could switch simultaneously. Others allow users to define which stimuli should be applied, that is, which circuit cells are going to be considered during simulation. Most of the time this definition is based on user experience and knowledge. However, both options may deteriorate the quality and the resulting accuracy of power grid simulation. A critical region may be neglected if the user misses the combination of grid stimuli that will cause the worst voltage drop or ground bounce (a false negative). Results from a simulation obtained on the assumption that all cells need to be

accounted for may also identify invalid critical regions of the power grid that are supposedly affected by voltage drop or ground bounce (a false positive). This occurs because in normal working conditions all cells in the circuit cannot draw current from the power grid at the same time. Moreover, this type of simulation may also increase total runtime and memory requirements from simulators. After this simulation procedure, the designer will try to solve IR-drop problems, usually by placing decoupling capacitance inside those critical regions. If those regions are noncritical, from a voltage drop and ground bounce point of view, the insertion of decoupling capacitance will only increase the overall static power consumption and it will be a waste of silicon area. These circuit changes can themselves cause voltage drop and ground bounce to appear in other circuit regions.

As mentioned previously, commercial tools in use nowadays resort to a variety of different methods, including vector-driven and vector-free (also referred to as vectorless) techniques for input stimuli generation. In vector-driven methods, the user or some external tool typically provides the stimuli, as described earlier. Some tools, such as *Quartz Rail* from Magma Design Automation, support vector-free transient analysis, automatically deriving internal activities within the timing windows by running a Monte Carlo simulation process based on boundary conditions and activity propagation. Sequence Design's *CoolTime* product also boasts a vectorless algorithm that is supposed to create a realistic design stimulus predicting the worst case. Details of the procedure, however, are scarce. Others tools, such as Cadence Design Systems' VoltageStorm, also support vector-free techniques but compute the stimuli based on analysis of the timing behavior of the circuit, together with estimates of cell switching activity and additional information about the circuit and the cell switching characteristics. As an industry leader in this sector, Apache Design Solutions also boasts a proprietary algorithm that is supposedly able to determine worst-case behavior. According to public information, it precomputes switching patterns for cells, which it uses together with cell switching estimates and physical design information. In reality neither technique can really guarantee exact worst-case analysis, so they tend to err in the side of caution, leading most of the time to pessimistic design constraints.

3. AUTOMATIC STIMULI GENERATION - PROPOSED DESIGN FLOW

In this section we present our proposal for design flow targeted at automatic generation of input stimuli and leading to vectorless power grid analysis. Besides automating the procedure, the goal of our proposal is also to ensure that the selected inputs correspond to relevant, realistic worst-case conditions for the grid operation. At the heart of this proposal is the realization that additional information is available which could be used to improve our selection of the stimuli for power grid simulation.

This proposed flow, which incorporates the proposed processing for input stimuli generation, is presented in Figure 3. Strictly speaking the flow described is not necessarily different from that of Figure 2, but it shows additional detail about how the input generation process occurs. As shown in this

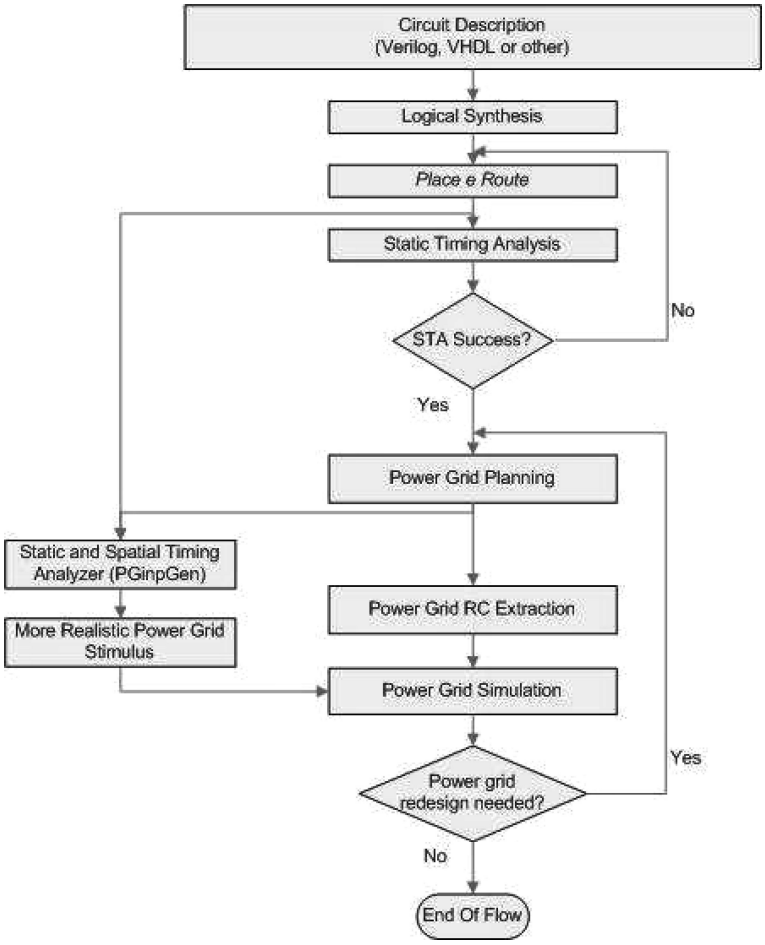

Fig. 3. Proposed design flow (with emphasis on power grid design).

figure, we introduce an additional task in the flow: the static and spatial timing analysis.

This task is where information both from physical design (circuit placement and routing) and timing is used to determine a reasonable realistic (worst-case) stimuli for simulation of power grids. Merging the spatial information available after cell placement and routing with information from power grid planning leads to spatial constraints about the current requirements imposed by the circuit to the grid. This information can then be crossed with the timing relationships that exist between cells, which are a consequence of circuit topology and cell delays and which can be easily obtained as a by-product (or added modifications) of Static Timing Analysis (STA). From this information, timing constraints are also available. Using both spatial and timing constraints, it is then possible to identify the space and time windows in which realistic circuit activity will lead to worst case loading of the grid. As we shall see, due to simplifications performed in the computational process, the input patterns generated

are not really guaranteed to correspond to the exact worst case, just the worst case given the information available and the assumptions made. In Section 7 we will validate the correctness of our estimates.

In general terms it is fair to say that the critical region of the power grid will be the determined by the existence of high levels of localized cell activity. Specifically, these *hotspots* will correspond to a location in the power grid where there is a high number of nearby cells all drawing current from the grid in a short period of time. In such a scenario, problems can occur if the switching of the cells demands from the power grid an amount of current which may not be available in that period of time. In this situation, the integrity of the power grid is affected, and the typical circuit behavior may also be jeopardized. Note that it is important not only that all cells draw current at "nearby" instants of time, but also that the current is drawn from the same "region" of the power grid. For example, consider the output of a cell with a fan-out of 7. When the output of the cell changes, 7 other cells might become active and draw current from the power grid. If these cells are placed far from each other, the power grid is likely to be able to handle this situation easily, which therefore does not cause much concern. The reasoning here is that assuming that the necessary current is available, the flow of that current will cause some voltage drop on the grid, but since currents are flowing to different locations, each will see just a small localized change and the overall behavior is not greatly affected. A different situation but with similar unimportant results occurs if we consider a chain of 10 blocks placed closer together. Assume that one logic change in the input to the first block will force all other blocks to become active in a sequence. Even though the blocks are all placed together and might draw current from the same branch of the grid, due to the propagation delay of each block, the maximum current drawn from the power grid does not occur simultaneously, and may not cause a significant problem to the power grid. Of course if the blocks are very fast (for instance, if each block is a minimum-sized inverter), then the combined switching occurs in a very short period of time and the grid may not have time to recover after each block/inverter switched. This may of course have an adverse effect on the grid potential.

It seems logical, therefore, then in order to determine scenarios of high concentration of nearby cells switching almost simultaneously, that information from the timing behavior of a circuit be generated and crossed with physical design information. For simplicity, this should be done in the context of a standard design flow and, as much as possible, using tools with which designers are to some extent familiar. To determine the time instants where each cell has the possibility of drawing current from the grid, namely, the instants where each cell may become active/switch, we can use an STA-like method. The placement information describes the location of each cell along the die. Information about the connection between each cell and the power grid is also readily available from the design data. Combining this timing information with the spatial information, obtained from the placement, allows us to determine which regions of the power grid may suffer the strongest impact in terms of drawn current in a short time span. Note that most of the information needed for the new task may already be available by the normal execution of the traditional flow

(or generated with simple modifications to it), but it is normally discarded or ignored in the standard power grid analysis tasks.

4. COMBINING TIMING AND SPATIAL INFORMATION

The static and spatial timing analyzer task uses information available after place and route operations to generate a more realistic set of simulation stimuli. To determine which devices may become active in a time frame, the circuit propagation delays are obtained using techniques similar to those employed by Static Timing Analysis (STA) tools. In order to determine the region of the power grid that is most affected by voltage drop or ground bounce, placement information is used.

4.1 Timing Information

Timing information regarding device activation enables to know the time frame where the largest number of devices switching can occur. In order to determine the time instants where each cell can become active, an operation similar to an STA is performed.

Traditional STA traverses the circuit computing the worst arrival time of all input signals for each cell. Since STA only considers the inverting or noninverting function of cells, this type of analysis is almost cell independent and can be done much faster than logic simulation. However, from the power grid analysis perspective, traditional STA cannot be used to determine the cells, switching activity, since it only keeps the maximum delay for each cell. Unfortunately, the worst case delay propagation may not coincide with the desired worst case switching scenario envisioned. In fact nothing prevents a significant number of cells from switching prior to their maximum delay and almost at the same time, causing a seemingly worst-case impact on the power grid. For example, consider a two-input AND gate with a propagation delay of $2ns$ and consider that one of the inputs can switch at time instant $2ns$ and the other at $5ns$. Typically for critical path detection, an STA tool would record the latest possible time at which the gate can change. However, we need to account for a possible output switch at instants $4ns$ and $7ns$, as each of these may propagate to other gates, leading to different switching activity. Therefore, we need to calculate and propagate all possible activation instants throughout the circuit, not only maximum delays.

Fortunately, computing all activation instants in a circuit can be done with some additional processing while traversing the circuit during STA. This operation may seem daunting since larger circuits will have a huge number of possible activation instants. However, in general most of the activation instants are close in time, so we can combine them into intervals with a relatively small loss of precision. In the case of the previously mentioned AND gate, we would say that the output could be active at both $4ns$ and $7ns$. Alternatively, if the time instants are too close, we can merge them into an interval and state that the gate can switch at any time during that interval. The creation of intervals is controlled by a parameter that can be changed by the user, trading precision for performance (speed and space). Figure 4 illustrates the procedure in a couple

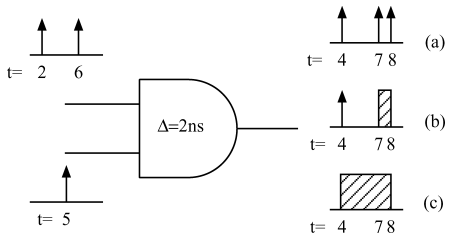

Fig. 4. Merging multiple activation instants. For the simple scenario given, three possibilities are shown for the output of a gate: (a) where all switching is transferred to the output; (b) where for instance it is assumed that the gate output requires at least 1 time unit (e.g., $1ns$) to stabilize (thus after one input changes at $t = 5$ it is not yet stable when the other input also changes at time $t = 6$); and (c) where for simplicity it is assumed that the output can change at any time since the arrival of the first transition and the end of the last one, but the exact switching instants are not kept.

of simplified settings using the previously mentioned scenario but adding an additional transition to the first input at time $t = 6ns$. If two switching instants happen within a very small interval, merging them can also be seen as a way of directly modeling inertia in the circuit: since the energy transfer associated with a node switching cannot happen instantaneously, multiple switching of a node within a very short interval is not physically acceptable; in this case we merge the two events and simply record the interval in which switching may occur. In the general case, the merging of switching instants can also be done in order to reduce the list of possible activity instants, hence reducing the computational complexity of the method at the cost of loss of accuracy, since now the exact switching instants are not known precisely.

Having determined all possible activation instants/intervals along the circuit, we can determine the worst timing window (that is, the window with the largest number of cells with the possibility of switching, or becoming active). Computationally this is achieved by determining the maximum number of intervals that fall inside a sliding window with a user-defined size (an appropriate size is dependent on technological parameters dictating, for instance, grid recovery time, switching frequencies, etc.). To compute this we simply need to maintain information regarding the time interval within which each cell can become active. Dividing real time into several virtual timing windows, it is then possible to compute the time frame where there is the largest number of active cells. The window sliding mechanism is emulated in a discrete sense by allowing the time window to slide along the time axis. User-defined parameters determine the window size and control the sliding mechanism, allowing to trade accuracy for performance. For reasons of efficiency, sliding of the window is also controlled such that at least one new activation enters or exits the window when it slides; that is, if there are no events for a certain period, the window is quickly moved forward. This is in essence akin to event-driven simulation and allows to slide the window faster for periods when the circuit activity is scarce.

4.2 Spatial Information

As mentioned in Section 3, the worst case situation from a power perspective corresponds to having a significant number of devices becoming active and

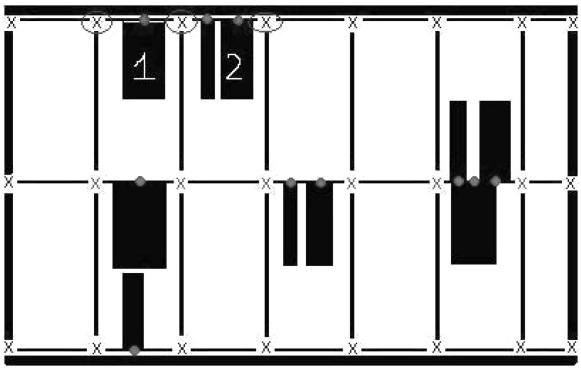

Fig. 5. Discrete grid model for power lines.

drawing current from close regions of the power grid. Accounting for this locality factor introduces the necessity of not only having timing but also spatial information, namely the spatial information describing the location of the cells in the design as well as the connections between the devices and the power grid. To account for the impact that active cells have upon the power grid, we create a discrete model of the circuit power grid. This model is composed by grid points that result from the intersection of the several power grid structures (intersections between the outer rings, the grid stripes, and the standard cell wires in standard cell technologies). Figure 5 shows a mock-up power grid and the corresponding discretized grid points marked with an $\times$.

For simplicity of the description assume that there is a unique point connecting the cell to the power grid (in fact in many situations standard cells are connected to the power grid through a small region of abutment). This unique feeding point, for each cell, is located in the middle of the abutment region and is shown in Figure 5 by a round point at the top/bottom of each cell. It is also assumed that each cell that becomes active impacts the discrete power grid at only two points. These two grid points are on the left and right side of the cell feeding point. Note that the cell position between these two points is also important, since the closer the cell is to one of the points, the higher the impact it causes over that particular point and the lesser the impact caused over the other. From Figure 5 it is easy to see that for example cells 1 and 2 (top leftmost), both contribute to the same grid point between them (in general all cells will contribute to the same grid points as their neighbors in the same row). So, if these two cells are active, the impact caused in this grid point is higher than in any other. These considerations are of course not essential to the procedure and if we consider cells to be connected through abutment, a similar reasoning would indicate which grid points should be considered sensitive from that cell's viewpoint, namely all that are nearby or within the abutment region.

In order to determine the region with the most impact, a spatial window with user-defined dimensions is moved along the die. In each window position, all the grid points covered by that spatial window are considered in order to identify the region that has suffered the most impact from active cells. This is illustrated in Figure 6. Crossing this spatial information with the time instants/

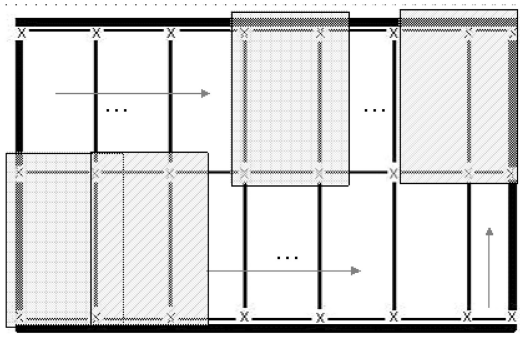

Fig. 6. Depiction of mock-up power grid and spatial window processing.

intervals, determined in the previous section, we are able to identify the worst timing/spatial region from the power grid perspective.

5. THE PGinpGen TOOL

In this section we describe a software tool that was developed to implement the proposed ideas for automatic generation of realistic power-grid input stimuli. To determine which devices may become active in a time frame, the circuit propagation delays are obtained using techniques similar to those employed by static timing analysis (STA) tools and generate information about the time intervals within which cells can become active. In order to determine the region of the power grid that is most affected by voltage drop or ground bounce, placement information is used. To that end, a new tool PGinpGen has been developed and integrated into a standard flow. In the next sections we describe the models and the algorithms used to compute the timing and spatial information and how both are combined by the PGinpGen tool to identify the cells and the corresponding region of the power grid in which a severe voltage drop or ground bounce may occur. A complexity analysis of the algorithm proposed is also included.

5.1 Algorithmic Description

A software tool that implements the additional task described in Figure 3 was created and named PGinpGen (for Power Grid INPput Generation). It was written in C/C++ for efficiency, was built over OpenAccess Gear Timer [Xiu et al. 2005] and integrated in the Cadence environment. This tool receives the circuit information from the OpenAccess database, reads the technology library, `.lib` file, and accepts constraints from a `.sdc` file. PGinpGen then generates a report describing the result of the analysis of the circuit power grid.

A pictorial description of the flow of the algorithm implemented by PGinpGen can be seen in Figure 7. The algorithm has the following steps.

(1) Create a discretized model of the circuit power grid using physical information (e.g., from the OpenAccess database).
(2) Determine the time instants at which each cell has the possibility of becoming active or inactive (modified static timing analysis with recording of

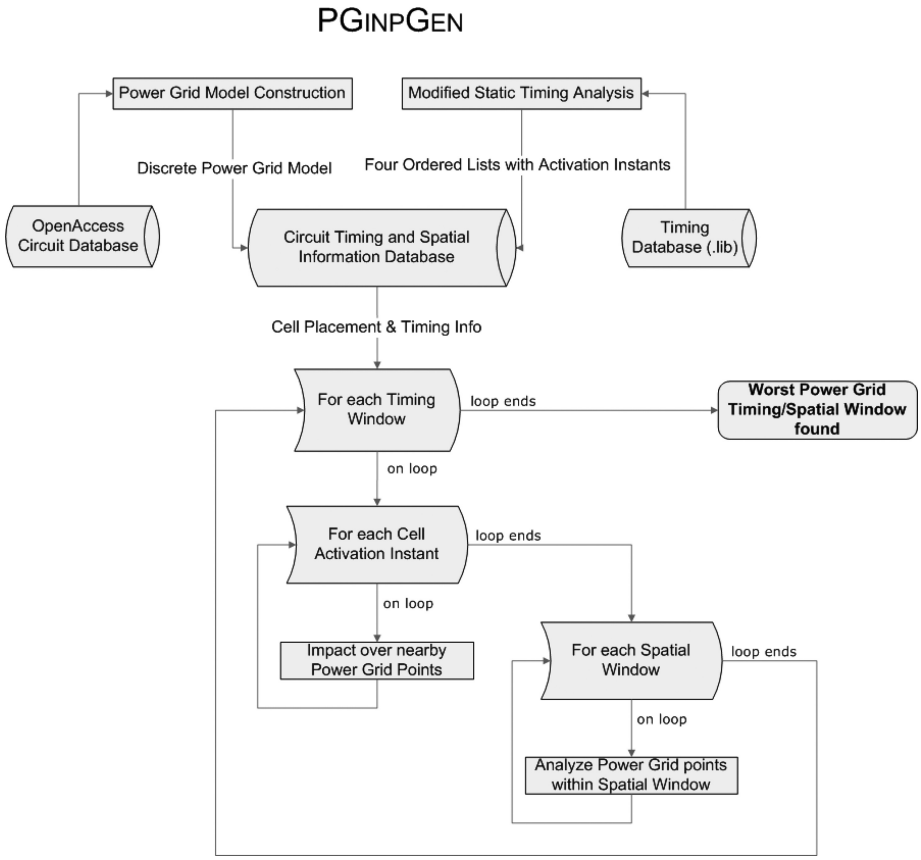

Fig. 7. Flowchart description of PGinpGen.

activation intervals), for each type of output transition (rise and fall). To this end, PGinpGen uses information from the .lib library characterization file to calculate the specific delay for each cell and uses an STA traversal algorithm to compute all the relevant instants/intervals.

To improve performance, the instants/intervals are placed in four linked lists, where two of the lists contain the minimum and maximum values of the rise transitions, and the other two the fall transition information. The first list of each type of transition allows us to know when each cell can become active.

(3) Sort, in ascending order of time, all four mentioned lists of instants obtained in the previous step.

(4) Determine the current timing window. Use the sorted information to know which cells may be active within this timing window. This is done incrementally within the lists by analyzing only those cells that changed their active/inactive status.

(5) For each cell that becomes active during a timing window, its impact over the grid points is added (incremented). For each cell that becomes inactive

during this timing window, its impact over the two closer grid points is subtracted (decremented).

(6) For the current timing window, the power grid is analyzed using a spatial window. While this spatial window moves along the die, the impact of active cells over the power grid points inside the window is observed. For a contribution with a maximum impact, the information about the corresponding timing and spatial window is updated.

(7) While not all activation instants have been considered, make the timing window slide through time and go to step 4.

(8) As a final result PGinpGen determines which timing window has the greatest impact over a certain region of the power grid (worst power grid timing/spatial window). It also knows which cells are responsible for this situation (e.g., are active within this timing and spatial grid), which can be useful for the designer.

Note that PGinpGen reads the netlist and placement information from the OpenAccess database. So, before running the tool, the design must have been previously placed and routed, and must have a regular power grid defined. Generating the associated stimuli for simulation simply requires translating each possible cell switching within the given interval into an appropriated stimulus for the simulator. This can be used using data typically available or generated from cell precharacterization.

5.2 Complexity Analysis

This section analyzes the computational costs for the algorithm implemented in the PGinpGen tool.

PGinpGen starts by reading the power grid description from the OpenAccess database. It converts the physical description into a discretized model, as mentioned in step 1 of the algorithm described on the previous subsection. All nodes of the discretized model result from the intersection of the multiple power grid physical structures. Therefore this operation takes a runtime that is proportional to the number of discretized points. We have

$$O(\#stripes \cdot \#standard\ CellWires), \tag{1}$$

where #*stripes* is the number of vertical power grid lines and #*standard CellWires* the number of horizontal power grid lines, according to the orientation shown in Figure 5.

Circuit traversal and creation of time instants/intervals for each cell (step 2) can be estimated as

$$O(\#cells), \tag{2}$$

where #*cells* is the number of logic gates in the circuit. Note that each cell is visited only once per traversal and we are considering that the number of intervals processed per cell is limited by a constant number (in particular if we consider the merging of instants/intervals as described before). This is clearly an approximation, but is a fairly acceptable one assuming the circuit was well designed.

The sorting operation over the four rise and fall transition lists (step 3) is done using quicksort and takes (in the average case)

$$O(\#intervals \cdot \log(\#intervals)), \tag{3}$$

where #*intervals* is the total number of intervals that need to be sorted, which given the preceding approximation, is proportional to the number of cells in the circuit.

The intervals processing time (steps 4 to 6) is proportional to

$$O(\#timingWindows \cdot \#spatialWindows), \tag{4}$$

where #*timingWindows* and #*spatialWindows* are the number of timing and spatial windows that will be analyzed.

Note that the user can influence steps 2, 4, and 6 by changing some PGinpGen configuration parameters. However, considering all steps of the algorithm, the time PGinpGen takes to analyze a single circuit is bound by the sorting operation and is proportional to

$$O(\#cells \cdot \log(\#cells)). \tag{5}$$

Observe that the proposed algorithm has a small computation overhead when compared with traditional STA algorithms, that typically runs in time proportional to $O(\#cells)$.

6. PRACTICAL CONSIDERATIONS

In this section we discuss some practical issues regarding the properties of the generated input stimuli, the robustness of the procedure, and additional uses of the approach and tool.

6.1 Exactness Considerations

The combination of spatial and timing information as described in Section 3 is intuitively a powerful mechanism for detecting a set of worst-case locations on the grid from the standpoint of grid behavior and node voltage drop. The procedure outlined in Section 4 consists of an algorithm to effectively combine such information, hopefully leading to a worst-case input setting. However, our ability to actually generate the exact worst-case setting is limited for a variety of reasons. The first reason is that the time-domain switching information is generated through Static Timing Analysis (STA). STA is an efficient procedure to perform timing analysis on a circuit but is known in general to provide only an upper bound on the exact timing of the circuit. One of the reasons for this is that most STA procedures perform conservative computations, assuming for instance that signal delay through a gate is a function of the latest arriving input. This does not take into account the logic values on those inputs, which may fail to lead to gate switching, and furthermore false paths are also usually not accounted for. Also, since for simplicity we may group transitions together in the same interval, some information may be lost. Weighting the intervals in a manner that estimates the switching activity inside them may minimize

the underlying error. An additional limitation has to do with the windowing procedure adopted. Obviously, for practical reasons, the sliding of the time and spatial windows, as described in Section 5, is not continuous. Therefore, from a practical standpoint we really do not account for all possible time and spatial windows. Theoretically, in the limit, it is possible to circumvent this limitation by making the window sliding as small as possible, but in practice this is not desirable. As a consequence, the results obtained with the proposed procedure are not guaranteed to correspond to the exact worst-case behavior, but are always realistic scenarios that stress the power grid capabilities.

In Mangassarian et al. [2007], the authors propose an exact procedure for determining the worst-case input stimuli. The procedure performs a symbolic simulation of the underlying digital network, implicitly generating a much larger circuit with time and switching information associated and then determines the exact input combination that leads to maximum switching. This procedure is computationally very costly and in practice thoroughly unusable for full chip analysis. A simplified version with lower computational complexity that uses the concepts of time and spatial windows in a manner similar to those described in this article is independently pursued in Morgado et al. [2008].

Even though, for the reasons mentioned, we cannot always guarantee that the exact worst case setting is detected, in practice the procedure proposed is able to automatically detect realistic worst-case settings which consist of very relevant input stimuli for power grid simulation.

6.2 The Effects of Variability

As feature sizes decrease to nanometer scale, the impact of process parameter variations in circuit performance becomes extremely relevant. As a consequence, much attention has been recently devoted to this issue in order to model and estimate the effects of variability on circuit design and performance. As an example, in the timing arena, Statistical Static Timing Analysis (SSTA) has been introduced as a form of incorporating variability effects in traditional static timing analysis [Liou et al. 2001; Visweswariah et al. 2004] but its applicability has been questioned, as its usage could ultimately entail an overhaul of the timing verification flow [Najm 2005].

Given that the proposed procedure combines information from timing analysis of the circuit with placement information for the various devices as well as the power grid connections, it stands to reason that parameter variability could have an effect on the results generated. While a precise analysis of these effects is beyond the scope of this article, it is worthwhile to ponder the relevance and magnitude of such effects. Clearly since parameter variability has potentially a first order effect on timing and since our proposed approach depends on timing information from gate switching, this effect should be taken into account. However, we point out that our approach does not need nor use precise timing information from gate switching, rather it relies on information regarding intervals of possible activation. In fact, since the procedure merely detects clusters of high switching activity in some time window, the precise timing information is not a first order concern. Therefore, the influence of parameter variability of

the switching characteristics of the circuit can, from our standpoint, perhaps be neglected.

Parameter variability can, however, affect power grid behavior in other ways. For instance in Ferzli and Najm [2003a, 2003b], the issue of increased device leakage due to variations in device's V_{th} is examined and its effect on power grid behavior is discussed. Such an effect is mostly a static effect and the corresponding analysis complements the one that underlined the approach described here. The effects of variability on dynamic power grid behavior have also been studied, for instance, in Ghanta et al. [2005] and Ghanta and Vrudhula [2007]. Such analysis involves the simulation of a stochastic representation of the power grid. Such a simulation requires stimuli and therefore the approach pursued here is again complementary to that study. Still, a more complete study and formulation of the full effects of variability in power grid analysis is clearly desirable.

6.3 Hotspot Detection

While the main goal of PGinpGen is to to be used during sign-off for input stimuli generation, the procedure can also be used for hotspot detection. In fact, analysis of the global switching activity generated by the tool allows to detect "activity hotspots" in the floorplan of the design and to act accordingly. This is interesting given that the ensuing power grid analysis is quite expensive and resource consuming, and can be obviated (in reality delayed until fixes are introduced) in certain settings.

The tool can also be used during the design phase using incomplete data and possibly a mock-up grid, to determine if certain regions have excessive localized activity and may require the addition of a decoupling capacitor. Clearly, given the incomplete and temporary nature of the information available in such stages, the results of the tool should be regarded as advisory and properly interpreted in that context. Fixing a potential problem at some early stage of the design does not guarantee robustness of the grid at that location since only very incomplete information is available for the analysis.

7. RESULTS

In this section, we present results obtained with PGinpGen in order to characterize the tool and to validate the results obtained. The first set of experiments are aimed at describing the tool's behavior and results. In a second set of experiments we attempt to validate the results obtained by comparisons with two additional techniques. In order to perform the tests to be described, the tool was integrated in a standard framework from Cadence Design Systems. All design data is available from an `OpenAccess` database to which the tool interfaces directly and where results are saved.

7.1 Tool Characterization

In order to demonstrate the functionality and behavior of the tool, we tested PGinpGen over 7 benchmark circuits from ISCAS'89 (these circuits were mapped into a standard cell library with three types of cells, namely AND,

Table I. PGinpGen vs Normal STA Runtimes

Circuit	# *Cells*	*STA* (*s*)	PGinpGen (s)
s27	23	0.15	0.16
s1196	945	0.70	0.87
s5378	2680	2.47	3.08
s13207	6084	14.83	16.37
s35932	24732	139.59	142.33
s38417	20872	77.19	82.02
s38584	27150	68.17	73.90

Table II. Critical Region for each of the Benchmark Circuits

Circuit name	*Grid size*	*Worst timing/spatial wind.*		*Active Cells*
		Time (*ns*)	*Region*	
s27	[5,2]	1.8 to 1.9	[2,1] to [2,2]	43%
s1196	[14,14]	3.3 to 3.4	[5,0] to [5,4]	7%
s5378	[28,31]	4.6 to 4.7	[4,18] to [6,27]	8%
s13207	[43,51]	6.7 to 6.8	[19,34] to [22,51]	10%
s35932	[92,85]	2.1 to 2.2	[43,44] to [70,72]	18%
s38417	[93,85]	8.8 to 8.9	[57,58] to [84,85]	22%
s38584	[95,85]	2.9 to 3.0	[59,15] to [87,43]	15%

NOT, and D-type FlipFlop). Table I presents the number of cells of each benchmark (# *Cells*) and the runtimes of the OpenAccess Gear Timer Static Timing Analyzer (*STA*) and our tool (PGinpGen), when using a timing window size of 0.1*ns* and a timing increment of 0.01*ns* (the latter is used to discretize real time). The results were obtained on a Linux-based Pentium IV at 3Ghz, with 1GB of RAM memory. The size of the spatial window was chosen for each circuit in order to have an approximate total of 40 spatial windows covering the die. As expected, we conclude from Table I that the modification of the static timing analysis procedure along with the use of spatial information has a small impact regarding runtime.

For each circuit, the worst timing/spatial window detected is shown in Table II. The *Grid size* column indicates the upper-right indexes of the discretized grid model, the lower-left indexes being [0,0] for all circuits. The *Interval* and *Region* columns indicate the critical timing/spatial region identified by the tool. Finally, the *Active Cells* column refers to the maximum percentage of active devices inside the critical region. Note that in general only a small percentage of cells (less then 25% for larger circuits) are responsible for the worst case scenario.

For the purpose of illustration we presented and discuss the analysis results for the smaller circuit (s27). In Figure 8 the logical description of the s27 circuit is shown. After analyzing the s27 circuit power grid, the critical spatial/timing window is shown in Figure 9.

As can be seen by comparing Figure 8 and Figure 9, almost all the cells that are detected by PGinpGen as capable of becoming active in the critical region are closely positioned in space (inside the spatial window) and in time (the cells are connected to each other). The cells that lay inside the critical region are

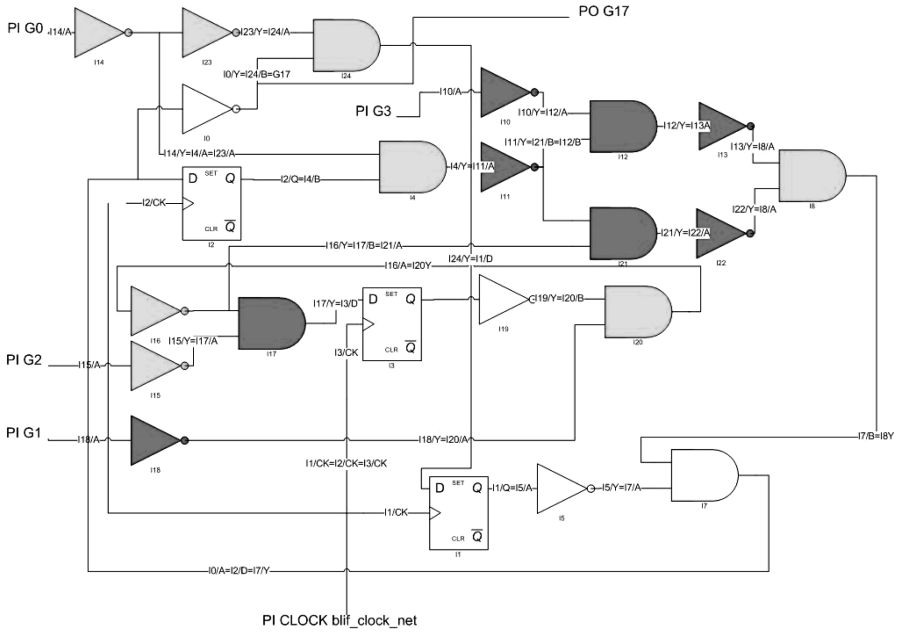

Fig. 8. Logical description of the mapped s27 circuit.

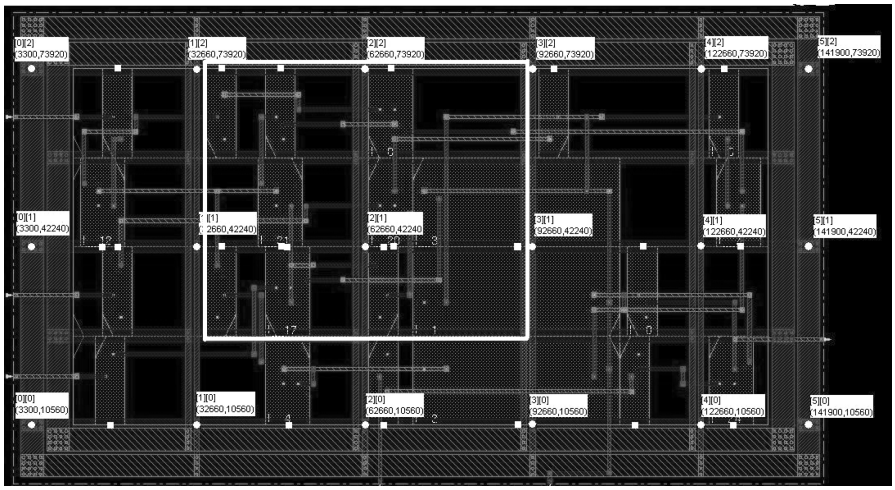

Fig. 9. Critical region in s27 circuit power grid.

represented in a darker color in Figure 8. Only the colored cells may be active inside the timing window considered (1.8*ns* to 1.9*ns*).

Finally, Figure 10 shows a graphical depiction of the switching activity of another example circuit. Visualized as a color plot, such a plot allows the identification of the areas of the floorplan that exhibit higher switching activity. This type of qualitative information can be used for a rough, quick hotspot detection, as mentioned previously.

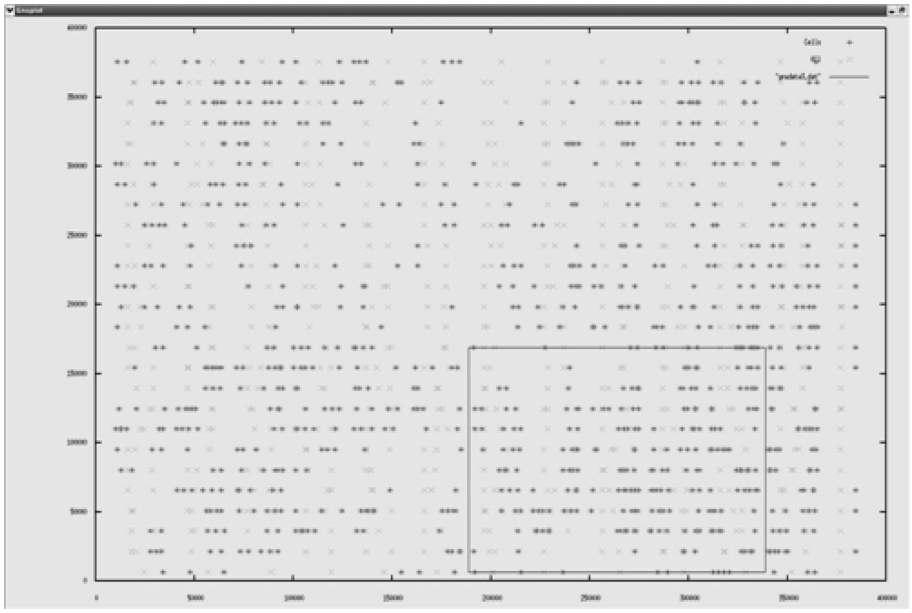

Fig. 10. Full-chip pattern of switching activity.

7.2 Tool Validation

For a tool such as PGinpGen that attempts to generate realistic stimuli for accurate power grid analysis, two considerations are of the utmost importance: First, the tool should, as much as possible, attempt to identify the worst-case switching scenario for the design; second, the tool should not underestimate the switching activity of the circuit, thus missing a critical scenario. In other words, and given the approximations done during processing of the timing and spatial windows' information, the tool should identify the worst case scenario, potentially outputting a set of stimuli that may lead to a slightly pessimistic analysis.

In view of these goals and in order to validate the results obtained with PGinpGen we have devised the following procedure. For a number of example benchmarks, we have computed three estimates of switching activity.

(1) We have computed a direct approximation to the exact worst case switching scenario. This is accomplished by running Monte Carlo simulations of the logic design, capturing the switching information, crossing with spatial information, and determining the worst spatial+timing window from the perspective of switching activity. We used a modified version of SIS for this task.
(2) We have computed the stimuli generated for vectorless power grid analysis by a commercial tool, namely VoltageStorm, from Cadence Design Systems. This information is then processed to determine the worst case spatial+timing window.
(3) Finally, we have used PGinpGen to determine our own estimate of the switching behavior of the circuit.

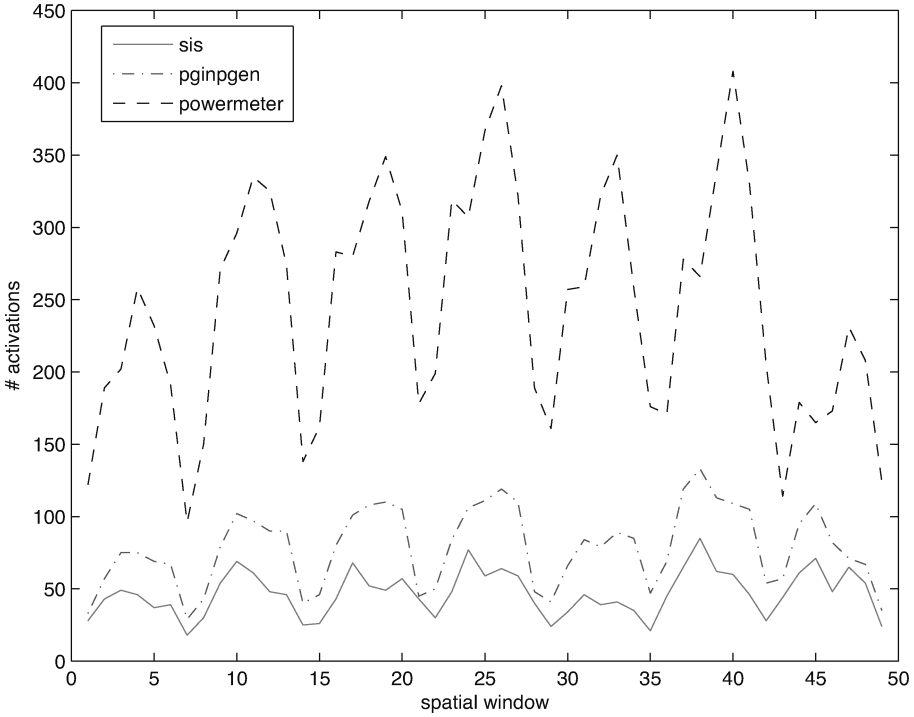

Fig. 11. Comparison of switching activity estimates for circuit c432.

For the purposes of illustration and to simplify the description, in the following we will show results from a single timing window (large enough to contain all circuit activity) and assume a mock-up power grid where 49 spatial windows are superimposed (this corresponds to a 4×4 griding of the layout and sliding the spatial window by half its size each time).

Figures 11 and 12 show plots with switching activity estimates for two circuits from the ISCAS'85 benchmark suite. For each circuit, the plot shows the number of cells in each spatial window that switch during the timing window considered (here chosen as the critical delay of the circuit). The plots shown are, in order, `sis`, which indicates the results of a Monte Carlo logic simulation of the circuit where the number of cell transitions is recorded and then crossed with spatial information, `pginpgen` is the result computed with PGinpGen, and finally `powermeter` is the estimate computed with PowerMeter, a tool used inside VoltageStorm.

The results depicted in the plots should be regarded as illustrative and not necessarily as definite proof of any particular characteristic of any of the algorithms shown. However, the results shown are quite interesting in that in both cases all curves show similar overall behavior, which seems to indicate that, up to different accuracy levels, they seem to be measuring similar properties. The `sis` curve is assumed to indicate the exact switching activity of each timing window (albeit only 10000 Monte Carlo vectors were used). As such, it should be considered as the more accurate pattern. Interestingly enough `pginpgen` shows

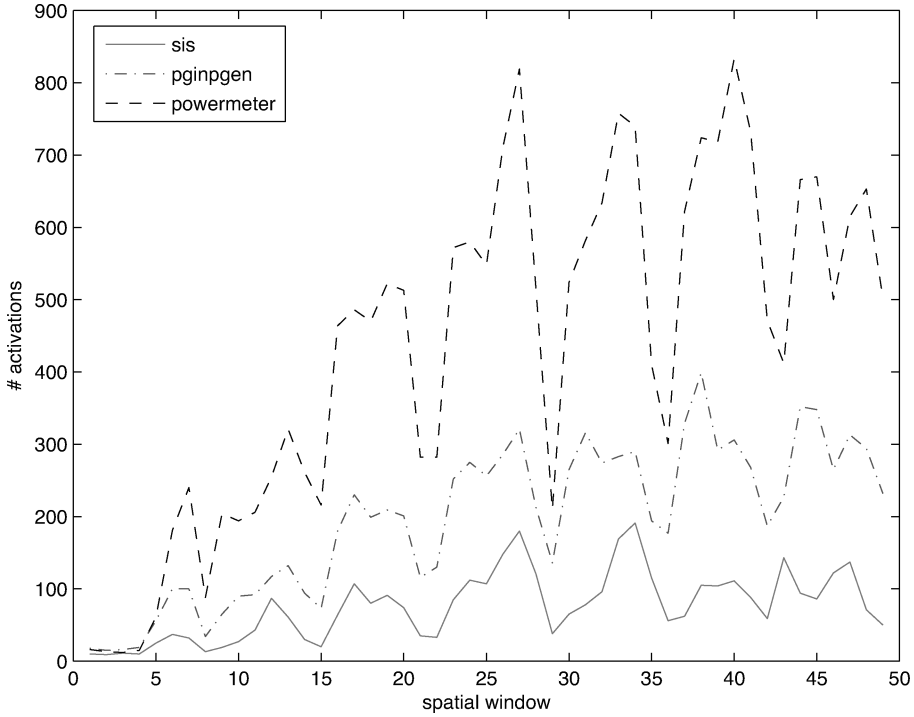

Fig. 12. Comparison of switching activity estimates for circuit c2670.

a pattern of switching very similar to both sis and powermeter but appears to be less pessimistic in estimating the number of activations in each spatial window. While there is no proof that this behavior always happens, we consistently saw it on the examples we ran. A more detailed analysis, however, shows some differences. For instance, for c432, sis and pginpgen seem to indicate that the highest switching activity is inside window no. 38. However, powermeter seems to indicate windows no.40 as the one with highest activation count. For c2670, sis pinpoints windows no.4, while pginpgen and powermeter indicate, respectively, windows no.38 and no.40, clearly overestimating the activity in some windows which would be then considered as critical.

The fact that all methods show similar activation patterns is in some sense a form of validation for pginpgen. However, the results also indicate that caution should be taken in analyzing the results and that perhaps several of the windows with highest activation count should be considered as potentially the worst case scenarios. Input patterns for such cases should be generated and used to simulate the grid.

8. CONCLUSIONS

In this article we described a software tool, PGinpGen, that generates a reasonable, realistic set of stimuli for simulation. The approach proposed uses placement and netlist information to compute the stimuli for a power grid simulation flow. The resulting stimuli indicate that only a fraction of the gates change in

any given timing window, leading to a more robust verification methodology, especially in the dynamic case. Generating such stimuli is akin to performing a standard static timing analysis, so the tool fits well within conventional design frameworks. Furthermore, the tool can be used for hotspot detection in early design stages. Comparison of the tool's output with an approximation to the exact switching activity and to the result of another commercial tool indicates that PGinpGen is a competitive and efficient alternative for generating realistic stimuli for power grid analysis.

ACKNOWLEDGMENTS

The authors would like to thank P. Rasatarinera, from Cadence Design Systems, Inc., for his help with integration issues and results interpretation.

REFERENCES

CADENCE. 2001. Power grid verification. Whitepaper, Cadence Design Systems, Inc.

CHOI, S. H., PAUL, B. C., AND ROY, K. 2002. Dynamic noise analysis with capacitive and inductive coupling. In *Proceedings of the Asian and South-Pacific Design Automation Conference (ASP-DAC)*. ACM/IEEE, 65–70.

FERZLI, I. A. AND NAJM, F. N. 2003a. Statistical verification of power grids considering process-induced leakage current variations. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE Computer Society, 770–777.

FERZLI, I. A. AND NAJM, F. N. 2003b. Statistical verification of power grids considering within-die process variations. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*. ACM/IEEE Computer Society, 856–859.

FERZLI, I. A., NAJM, F. N., AND KRUSE, L. 2007. Early power grid verification under circuit current uncertainties. In *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 116–121.

GHANTA, P. AND VRUDHULA, S. 2007. Analysis of power supply noise in the presence of process variations. *IEEE Des. Test Comput. 24*, 3, 256–266.

GHANTA, P., VRUDHULA, S., PANDA, R., AND WANG, J. 2005. Stochastic power grid analysis considering process variations. In *Proceedings of the Design, Automation and Test in Europe Conference (DATE)*. ACM/IEEE, 964–969.

KOUROUSSIS, D. AND NAJM, F. N. 2003. A static pattern-independent technique for power grid voltage integrity verification. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*. ACM/IEEE, 99–104.

KOUROUSSIS, D. AND NAJM, I. A. F. F. N. 2005. Incremental partitioning-based vectorless power grid verification. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. ACM/IEEE, 358–364.

KRODEL, T. H. 1991. Powerplay-Fast dynamic power estimation based on logic simulation. In *Proceedings of the IEEE International Conference on Computer Design on VLSI in Computer and Processors (ICCD '91)*. IEEE Computer Society, 96–100.

LIN, S. AND CHANG, N. 2001. Challenges in power-ground integrity. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. ACM/IEEE, 651–654.

LIOU, J.-J., CHENG, K.-T., KUNDU, S., AND KRSTIC, A. 2001. Fast statistical timing analysis by probabilistic event propagation. In *Proceedings of the ACM/IEEE Design Automation Conference*. 661–666.

MANGASSARIAN, H., VENERIS, A., SAFARPOUR, S., NAJM, F. N., AND ABADIR, M. S. 2007. Maximum circuit activity estimation using pseudo-boolean satisfiability. In *Proceedings of the Design, Automation and Test in Europe Conference (DATE)*. IEEE, 1538–1543.

MORGADO, P. M., FLORES, P. F., MONTEIRO, J. C., AND SILVEIRA, L. M. 2008. Generating worst case stimuli for accurate power grid analysis. In *Proceedings of the PATMOS Workshop* (to be published by Springer in Lecture Notes for Computer Science). Springer-Verlag.

NAJM, F. N. 2005. On the need for statistical timing analysis. In *Proceedings of the ACM/IEEE Design Automation Conference*. 764–765.

NASSIF, S. R. AND KOZHAYA, J. N. 2000. Fast power grid simulation. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*. ACM/IEEE, 156–161.

SATO, T., SYLVESTER, D., CAO, Y., AND HU, C. 2000. Accurate in-situ measurement of peak noise and signal delay induced by interconnect coupling. In *Proceedings of the International Solid-State Circuits Conference Technical Papers (ISSCC)*. IEEE, 226–227.

SU, H., SAPATNEKAR, S. S., AND NASSIF, S. R. 2003. Optimal decoupling capacitor sizing and placement for standard-cell layout designs. *IEEE Trans. Comput.-Aided Des. Integrated Circ. 22*, 4, 428–436.

VISWESWARIAH, C., RAVINDRAN, K., KALAFALA, K., WALKER, S. G., AND NARAYAN, S. 2004. First-Order incremental block-based statistical timing analysis. In *Proceedings of the ACM/IEEE Design Automation Conference*, 331–336.

XIU, Z., PAPA, D. A., CHONG, P., ALBRECHT, C., KUEHLMANN, A., RUTENBAR, R. A., AND MARKOV, I. L. 2005. Early research experience with openaccess gear: An open source development environment for physical design. In *Proceedings of the International Symposium on Physical Design (ISPD '05)*. ACM Press, New York, 94–100.

ZHONG, Y. AND WONG, M. D. F. 2005. Fast algorithms for ir drop analysis in large power grid. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '05)*. IEEE Computer Society, 351–357.

Received September 2007; revised October 2008; accepted March 2009