HiSIM: Hierarchical Interconnect-Centric Circuit Simulator

Tsung-Hao Chen[1], Jeng-Liang Tsai[2], Charlie C.-P. Chen[3], Tanay Karnik[4]

[1] Synopsys, Inc.
[2] Electrical and Computer Engineering, University of Wisconsin-Madison
[3] Graduate Institute of Electronics Engineering, National Taiwan University
[4] Intel Corp.

ABSTRACT

To ensure the power and signal integrity of modern VLSI circuits, it is crucial to analyze huge amount of nonlinear devices together with enormous interconnect and even substrate parasitics to achieve the required accuracy. Neither traditional circuit simulation engines such as SPICE nor switch-level timing analysis algorithms are equipped to handle such a tremendous challenge in both efficiency and accuracy. In this paper, we establish a solid framework that simultaneously takes advantage of a novel hierarchical nonlinear circuit simulation algorithm and an advanced large-scale linear circuit simulation method using a new predictor-corrector algorithm. Under solid convergence and stability guarantees, our simulator, HiSIM, a hierarchical interconnect-centric circuit simulator, is capable of handling the post-layout RLKC power and signal integrity analysis task efficiently and accurately. Experimental results demonstrate over 180X speed up over the conventional flat simulation method with SPICE-level accuracy.

1. INTRODUCTION

In deep-sub-micron giga-hertz VLSI design, power and signal integrity has become crucial concerns other than performance. To accurately analyze the impacts of power fluctuation, capacitive, inductive, or even substrate coupling noise impacts, several decoupled static analysis algorithms have been proposed[1][2][3]. For example, to estimate power delivery fluctuations, [4][5] proposed to first characterize the gate current consumption by assuming an ideal voltage and then plug those PWL current waveforms into linear networks. However, albeit the efficiency of this approach, this algorithm may suffer accuracy and stability issues[6].

Second, to incorporate the noise impact to timing, several decoupled gate delay and interconnect noises analysis algorithm have been proposed. However, since the interconnect noise may impact the output behavior of gates, this algorithm also suffer inaccuracy issues. Moreover, this type of analyses may not be able to handle multiple-driver analysis tasks. Third, as for the timing window convergence issues, iterative window shrinking or enlarging algorithms have been proposed[7][8]. However, the static nature of this type of algorithms also imposes non-trivial pessimistic guard banding in into already tight timing requirement. With the diminishing timing and noise margin budgeting, the errors induced by those decoupled or static approximations may no longer be tolerable for modern VLSI designs. As a last resort, transistor-level simulators become the final means for sanity check.

However, the traditional transistor-level simulators such as SPICE[9] are not capable of handling such a large-scale computational expensive tasks since SPICE was developed in an era such that VLSI chips only contained a few transistors and the interconnect parasitic was not the dominating factor as well. Although the recent advancement of transistor-level simulation such as HSIM[10] already deployed hierarchical framework to take advantage of the spatial, temporary latency, and array structure such as memories to enhance simulation performance. However, the success of this type of algorithms may not be easily brought into the interconnect-dominated simulation cases such as full-chip power-delivery analysis especially taking not only coupling capacitance but also self and mutual inductance into consideration.

Recently, M. Zhao, et al[2] proposed to perform hierarchical analysis of power distribution networks. This method can only handle RC elements but not inductors and nonlinear elements due to asymmetricity of the system matrix. Another work, SILCA[11], used a semi-implicit scheme that takes advantage of interconnect dominate cases by using a modified chord methods. However, this algorithm may not perform well for the cases when there are a non-trivial number of transistors presented.

As a result, there is a lack of chip-level simulation algorithms, which can simultaneously take care of large-scale nonlinear and linear devices while maintaining both efficiency and accuracy. In this paper, we establish a solid framework to simultaneously take advantage of the novel hierarchical nonlinear circuit simulation algorithm and advanced large-scale linear circuit simulation method in a waveform relaxation manner using a novel predictor-corrector algorithm. Under solid convergence and stability guarantees, our simulator, HiSIM, a hierarchical and interconnect-centric circuit simulator, is capable of handling post-layout RLKC power and signal integrity analysis tasks efficiently

and accurately. Experimental results demonstrate that HiSim has over 180X speed up over the conventional flat simulation method with SPICE-level accuracy.

2. OVERVIEW OF NONLINEAR CIRCUIT SIMULATION

Transient analysis is applied to evaluate the large signal behavior of a linear/nonlinear circuit as a function of time. The transient analysis flow for non-linear circuits is shown in Figure 1(a). The DC solution is first calculated and serves as the initial condition (i.e. $t = 0$). In each time step, resistive models of energy storage elements such as capacitors and inductors are built using either backward Euler, forward Euler, trapezoidal, or multi-step approximation. Then a nonlinear system has to be solved to obtain the response for this time step. Keep increasing t until it reaches the time interval that is interested. Both DC and transient solutions are obtained by Newton-Raphson (NR) algorithm, which is shown in Figure 1(b). NR is an iterative method that generates a sequence that converges to the solution of a set of non-linear equations. In each iteration, it builds linear companion models of nonlinear devices by calculating the Jacobian at the previous solution or the initial guess, and solves the linear equations. Repeat the iteration until the sequence converges[12][13]. Several techniques such as iteration damping[14] and G_{min}-stepping[15] have been proposed and also implemented in this work to increase the converging rate and robustness of the NR iterations.

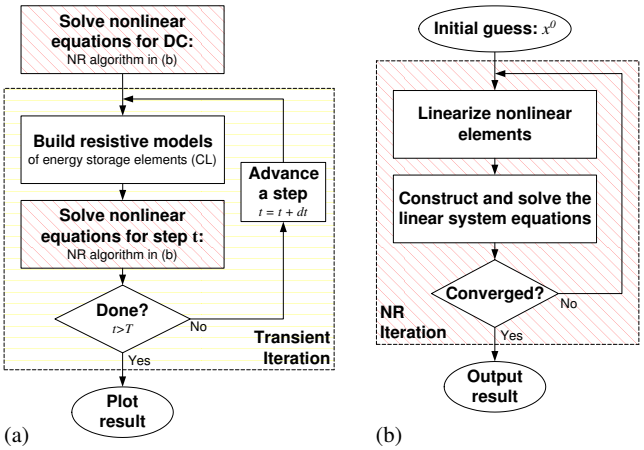

Figure 1: Simulation flow for circuits containing non-linear elements (a)Transient analysis flow diagram (b)The Newton-Raphson(NR) method

Each NR iteration contains one solution of a linear system, which means one Gaussian elimination or LU decomposition is required. Obviously, the computation effort of solving a linear system determines the runtime of the transient simulation. The complexity of the LU decomposition depends on the density of the matrix. It varies from super linear for a sparse system to $O(n^3)$ for a dense matrix. Due to the increasing complexity of VLSI designs and already important power and signal integrity issues, the circuits needed to be analyzed might be huge and with enormous number of nonlinear devices and linear elements (i.e. interconnect parasitic). Moreover, if the partial mutual inductance model[16] is included in the simulation, the long range inductive effect will cause a dense system matrix. Both these two reasons make traditional simulation engines such as SPICE hard to handle this kind of large-scale simulation tasks.

For interconnect-centric applications, huge system matrices that contain linear and nonlinear elements have to be factorized in each NR iteration. However, system matrices of two successive NR iterations only disagree on the entries corresponding to nonlinear elements (even only some of them) but not linear parts. Based on this observation, we develop a hierarchical simulation scheme that can greatly reduce the effort spending on solving linear systems and is presented in the rest of this paper.

3. HIERARCHICAL CIRCUIT ANALYSIS

In this section, we first propose the macro-modeling technique for a sub-circuit containing linear and nonlinear elements, and then describe the hierarchical scheme that assembles macro-models. We also discuss the performance gain of the hierarchical scheme compared to a conventional flat MNA-based simulation.

3.1 Macro-modeling Technique

Given a lumped circuit, its MNA system equation is as follows.

$$\mathbf{G}\mathbf{v} + \mathbf{C}\frac{d}{dt}\mathbf{v} = \mathbf{u} \quad . \tag{1}$$

In (1), $\mathbf{G}$ is the conductance matrix that is composed of the equations for resistors, conductors, independent voltage sources, and control sources, etc...; $\mathbf{C}$ is the susceptance matrix that consists of equations for energy storage elements such as capacitors and inductors; $\mathbf{v}$ is the vector of variables including nodal voltages and branch current variables that are necessary to introduce such as branch currents of independent voltage sources and inductors; $\mathbf{u}$ is the vector containing input current and voltage sources.

Using back-Euler approximation, transient analysis of Equation (1) is obtained:

$$\left(\mathbf{G} + \frac{1}{h}\mathbf{C}\right)\mathbf{v}^{j+1} = \left(\frac{1}{h}\mathbf{C}\right)\mathbf{v}^{j} + \mathbf{u}^{j+1} \quad , \tag{2}$$

in which h means the size of a time-step and the superscript j means the j^{th} time step. Let

$$\begin{aligned} \mathbf{A} &= \left(\mathbf{G} + \frac{1}{h}\mathbf{C}\right) \quad , \\ \mathbf{x} &= \mathbf{v}^{j+1} \quad , \text{ and} \\ \mathbf{b} &= \left(\frac{1}{h}\mathbf{C}\right)\mathbf{v}^{j} + \mathbf{u}^{j+1} \quad , \end{aligned} \tag{3}$$

and the system equation (2) can be represented in the following format:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad . \tag{4}$$

Similarly, forward-Euler, trapezoidal and other multi-step approximations can also be written in the format of (4).

For a sub-circuit shown in Figure 2, we can generate the macro-model for it by first grouping the nodal voltage and branch current variables into two parts, internal and external (port) variables. By reordering the system matrix $\mathbf{A}$,

the system equation (4) for this sub-circuit is rewritten as follows.

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_i \\ \mathbf{x}_e \end{bmatrix} = \begin{bmatrix} \mathbf{b}_i \\ \mathbf{b}_e + \mathbf{b}_o \end{bmatrix} \quad . \tag{5}$$

In (5), $\mathbf{x}_i$ and $\mathbf{x}_e$ are the vectors of internal and external variables respectively. Internal variables stand for those nodal voltages and branch currents that are not interactive with any components outside this sub-circuit. External variables include variables that are interactive with those outside this sub-circuit, such as port nodal voltages($\mathbf{v}_{n,e}$), inductance currents($\mathbf{i}_{l,e}$) coupled with external inductors, branch currents($\mathbf{i}_{c,e}$) required by control sources outside, and etc... $\mathbf{b}_i$ is the vector of current and voltage sources that connect to internal nodes. $\mathbf{b}_e$ is the vector of current sources($\mathbf{i}_{s,e}$) that connect to external nodes but belong to this sub-circuit. $\mathbf{b}_o$ is the vector of sources that are induced or controlled by the components outside, which include voltage drop caused by mutual inductors($\mathbf{v}_{l,o}$), control voltage($\mathbf{v}_{s,o}$) and current($\mathbf{i}_{s,o}$) sources, and etc... These variables are illustrated in Figure 2.

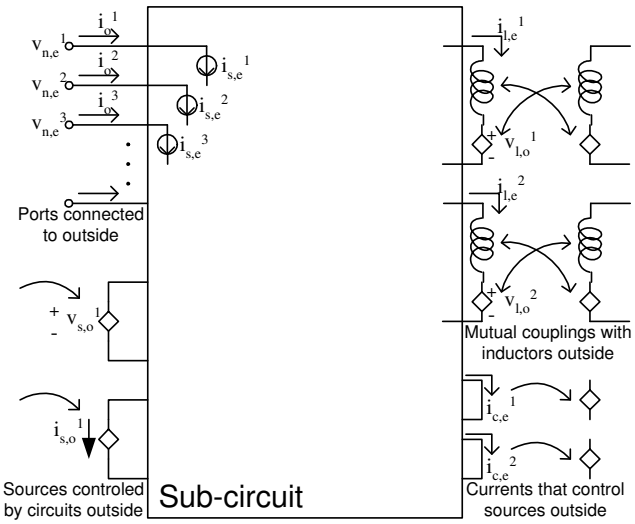

Figure 2: Illustration of ports in a sub-circuit

Rewriting the first set of equations in (5), we get

$$\mathbf{x}_i = \mathbf{A}_{11}^{-1} \left(\mathbf{b}_i - \mathbf{A}_{12} \mathbf{x}_e \right) \quad . \tag{6}$$

Substituting (6) into (5), we get the second set of equations as follows:

$$\mathbf{b}_o = \left(\mathbf{A}_{22} - \mathbf{A}_{21} \mathbf{A}_{11}^{-1} \mathbf{A}_{12} \right) \mathbf{x}_e - \left(\mathbf{b}_e - \mathbf{A}_{21} \mathbf{A}_{11}^{-1} \mathbf{b}_i \right) \quad . \tag{7}$$

We now factorize matrix $\mathbf{A}$ by block LU decomposition.

$$\begin{aligned} \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} &= \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{0} & \mathbf{U}_{22} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{L}_{11}\mathbf{U}_{11} & \mathbf{L}_{11}\mathbf{U}_{12} \\ \mathbf{L}_{21}\mathbf{U}_{11} & \mathbf{L}_{21}\mathbf{U}_{12} + \mathbf{L}_{22}\mathbf{U}_{22} \end{bmatrix} \end{aligned} \tag{8}$$

Let

$$\begin{aligned} \widetilde{\mathbf{A}} &= \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12} \\ &= \mathbf{L}_{21}\mathbf{U}_{12} + \mathbf{L}_{22}\mathbf{U}_{22} - \mathbf{L}_{21}\mathbf{U}_{11} \left(\mathbf{L}_{11}\mathbf{U}_{11} \right)^{-1} \mathbf{L}_{11}\mathbf{U}_{12} \\ &= \mathbf{L}_{21}\mathbf{U}_{12} + \mathbf{L}_{22}\mathbf{U}_{22} - \mathbf{L}_{21}\mathbf{U}_{11}\mathbf{U}_{11}^{-1}\mathbf{L}_{11}^{-1}\mathbf{L}_{11}\mathbf{U}_{12} \\ &= \mathbf{L}_{22}\mathbf{U}_{22} \quad , \end{aligned} \tag{9}$$

and

$$\begin{aligned} \widetilde{\mathbf{b}} &= \mathbf{b}_e - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{b}_i \\ &= \mathbf{b}_e - \mathbf{L}_{21}\mathbf{U}_{11} \left(\mathbf{L}_{11}\mathbf{U}_{11} \right)^{-1} \mathbf{b}_i \\ &= \mathbf{b}_e - \mathbf{L}_{21}\mathbf{U}_{11}\mathbf{U}_{11}^{-1}\mathbf{L}_{11}^{-1}\mathbf{b}_i \\ &= \mathbf{b}_e - \mathbf{L}_{21}\mathbf{L}_{11}^{-1}\mathbf{b}_i \quad . \end{aligned} \tag{10}$$

Thus from Equation (7), the equivalent equation of this sub-circuit can be expressed in the following format:

$$\widetilde{\mathbf{A}}\mathbf{x}_e - \mathbf{b}_o = \widetilde{\mathbf{b}} \quad . \tag{11}$$

The macro-model $\widetilde{\mathbf{A}}$ and equivalent sources $\widetilde{\mathbf{b}}$ can be calculated by (9) and (10).

3.2 Hierarchical Simulation

In the current design flow, virtually all circuits are hierarchical due to the complexity and design reusability. Simulation tools such as SPICE support sub-circuits but still use flat scheme to simulate. A sub-circuit such as a functional block usually contains large number of elements and internal nodes with relatively less number of ports. Therefore, we take advantage of this design hierarchy nature and attempt to earn efficiency during the dynamic simulation.

For a circuit containing one or several sub-circuits, we first build macro-models of all sub-circuits by (9) and (10). After all of them are in the condensed form, we can organize the whole system equation as follows:

$$\begin{bmatrix} \mathbf{A}_0 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{E}_0^T \\ \mathbf{0} & \widetilde{\mathbf{A}}_1 & \cdots & \mathbf{0} & \mathbf{E}_1^T \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \widetilde{\mathbf{A}}_n & \mathbf{E}_n^T \\ \mathbf{E}_0 & \mathbf{E}_1 & \cdots & \mathbf{E}_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{e0} \\ \mathbf{x}_{e1} \\ \vdots \\ \mathbf{x}_{en} \\ \mathbf{i}_{port} \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{b}}_0 \\ \widetilde{\mathbf{b}}_1 \\ \vdots \\ \widetilde{\mathbf{b}}_n \\ \mathbf{0} \end{bmatrix} \quad , \tag{12}$$

where $\mathbf{A}_0$ is the conductance matrix that is composed of elements not described in any sub-circuit, $\widetilde{\mathbf{A}}$'s are the macro-models of sub-circuits, $\mathbf{E}$'s are the adjacency matrices containing equations how ports of sub-circuits and global nodes are interactive, and $\mathbf{i}_{port}$ is the vector of current (or voltage if necessary) variables running through ports of sub-circuits. Equation (12) can be easily derived from (11) and is illustrated in the top layer of Figure 3.

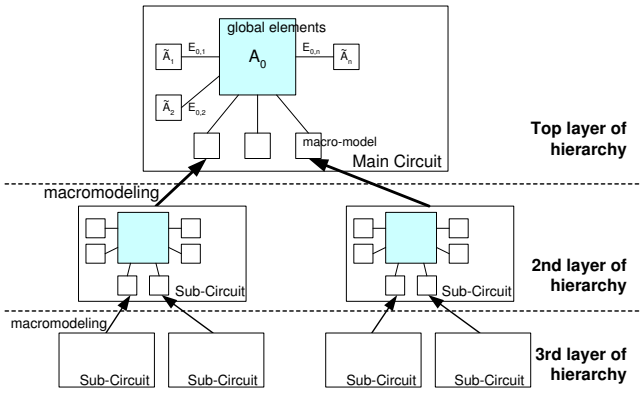

Figure 3: Circuit simulation with multi-layer of hierarchy

Note that Equation (12) represents only one level of hierarchy. While multiple levels are presented in the circuit

(as shown in Figure 3), the hierarchical macro-modeling can be done by a bottom-up search of the tree structure. We first create macro-models for the sub-circuits in the bottom, form the system equation by (12), and use (9) and (10) to construct models for the upper level of hierarchy. By repeating this procedure until the whole tree structure has been traversed once, we can get the equation of the top level of hierarchy.

3.3 Analysis of Computation Cost

We now analyze the computation cost of our hierarchical simulation scheme compared to the flat one. Assumed the cost of factorizing a matrix $\mathbf{a}$ is $\Gamma(\mathbf{a})$, the cost of the transient simulation using flat MNA solution is

$$k \times \Gamma(\mathbf{A}_N), \tag{13}$$

where k is the total number of NR iterations used to solve the transient response, and $\mathbf{A}$ is the flat MNA matrix whose subscript $_N$ means its dimension.

Supposed only one layer of hierarchy is used, and the cost to build a macro-model for a matrix $\mathbf{a}$ is $\Upsilon(\mathbf{a})$, the computation cost for the hierarchical simulation is

$$k \times \left(\sum_{i=1}^{n} \Upsilon(\mathbf{A}_i) \ + \ \Gamma \left(\mathbf{A}_0 + \sum_{i=1}^{n} \widetilde{\mathbf{A}}_i + \mathbf{E} \right) \right) \ . \tag{14}$$

The first summation in (14) is the total cost of building macro-models for all sub-circuits, and the second part, $\Gamma(\cdot)$, means the cost of factorizing the condensed system equation listed in (12). The difference between (13) and (14) is basically the difference between the following two different ordering methods: minimum fill-in and nested dissection. It is known that the latter is a little bit slower then the former. However, macro-model calculation can be executed in parallel, and the computation cost becomes

$$k \times \left(\mathbf{max} \left\{ \Upsilon(\mathbf{A}_i)|_{i=1\sim n} \right\} \ + \ \Gamma \left(\mathbf{A}_0 + \sum_{i=1}^{n} \widetilde{\mathbf{A}}_i + \mathbf{E} \right) \right) \ .$$

It is obvious that parallel processing can dramatically reduce the computation cost, so we do not address too much on it in this paper. The difficulty of parallel computing is that not every designer can access a parallel system. Therefore, we will show that even without parallel computing, our proposed method still greatly improves the performance of analyzing VLSI circuits with strong parasitic coupling effects.

First, to capture interconnect parasitic effects, huge amount of linear elements should be modeled and included in the analysis. Modules such as buses, power grids, clock trees, or even substrate parasitics can be isolated from nonlinear devices and become pure linear sub-circuits. There is no need to build macro-models for linear sub-circuits in each NR iteration. Linear macro-models are built at the beginning of the simulation if fixed-step simulation methods are used, and reused in the successive iterations. Thus, the computation cost becomes

$$\sum_{lin} \Upsilon(\mathbf{A}_i) + k \times \left(\sum_{non} \Upsilon(\mathbf{A}_i) \ + \ \Gamma \left(\mathbf{A}_0 + \sum_{i=1}^{n} \widetilde{\mathbf{A}}_i + \mathbf{E} \right) \right) \ . \tag{15}$$

From (15), the performance will be greatly improved if there exist many linear sub-circuits and if k is large. If multi-step methods are used, we only have to build those linear macro-models for every time step. Since several NR iterations are required to converge in each time step, the hierarchical simulation scheme still saves tremendous computational effort.

Second, even a sub-circuit contains nonlinear elements, it is not necessary to rebuild its macro-model in every NR iteration. A sub-circuit may be "quiet" while others are operating. This is so called temporal latency. We can check the port variables of a sub-circuit before we reconstruct its macro-model in each NR iteration. If the change of variables is under the given tolerance, computation cost of building model for this sub-circuit can be saved. For example, supposed the three curves in Figure 4 are the port responses of sub-circuits A, B, and C respectively. In region I, only sub-circuit A is activate; in region II, only B is changing. If we use conventional flat simulation scheme, we have to stamp and factorize the matrix with sub-circuits A, B, and C. The hierarchical scheme only has to construct the macro-model of A in region I, and B in region II.

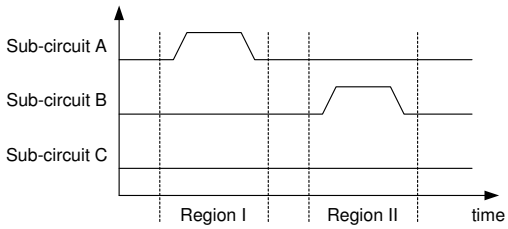

Figure 4: An example of temporal latency

Based on the above reasons, our hierarchical simulation scheme is competent to simulate large-scale interconnect-centric analysis. Runtime improvement of several different types of circuits will be reported in the result section.

4. PARTITIONED EXPLICIT METHOD

In this section, we propose a predictor-corrector algorithm that further improves the runtime of our hierarchical simulator in a waveform relaxation manner. Due to the high computation cost of LU decomposition, divide-and-conquer is usually used to improve the runtime of solving a problem. Some examples such as the ADI (Alternating Direction Implicit) method[17] and the waveform relaxation technique[18]. The ADI method is often used to efficiently solve problems with regular 2-D(or 3-D) grid structures. Instead of solving the whole 2-D(or 3-D) grid, it solves each direction separately, which reduces the complexity to linear, and shows good accuracy for some particular applications. Waveform relaxation algorithms break a system into pieces and solve them independently, which is usually used in circuit simulation to reduce the computation cost. Our proposed partitioned explicit method is also a divide-and-conquer scheme, which partitions the circuit into linear and nonlinear parts and performs LU to them separately.

4.1 Explicit Predictor for NR Iteration

We first group the circuit into two different sets, sub-circuits with and without non-linear elements. Thus, the system equation (12) can be written as follows.

$$\begin{bmatrix} \mathbf{A}_n & \mathbf{0} & \mathbf{E}_n^T \\ \mathbf{0} & \mathbf{A}_l & \mathbf{E}_l^T \\ \mathbf{E}_n & \mathbf{E}_l & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_n \\ \mathbf{x}_l \\ \mathbf{i}_{port} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_n \\ \mathbf{b}_l \\ \mathbf{0} \end{bmatrix} \ . \tag{16}$$

$\mathbf{A}$, $\mathbf{x}$ and $\mathbf{b}$ are the system matrix, the vector of unknown variables, and the right-hand-side vectors respectively, which are defined in (3). $\mathbf{E}$ and $\mathbf{i}_{port}$ are the adjacency matrix and the vector of currents running through the connection similar to those in (12). The subscripts n and l represent sub-circuits containing nonlinear elements and those containing only linear elements respectively.

By rearranging the terms, we split Equation (16) into two parts.

$$\begin{bmatrix} \mathbf{A}_n & \mathbf{E}_n^T \\ \mathbf{E}_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_n \\ \mathbf{i}_{port} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_n \\ -\mathbf{E}_l \mathbf{x}_l \end{bmatrix} \tag{17}$$

$$\mathbf{A}_l \mathbf{x}_l = \mathbf{b}_l - \mathbf{E}_l^T \mathbf{i}_{port} \tag{18}$$

By using the relation in equations (17) and (18), we derive our partitioned explicit method to reduce the computation cost of each NR iteration. We first set

$$\mathbf{i}_{port}^{j+1(0)} = 2\mathbf{i}_{port}^{j} - \mathbf{i}_{port}^{j-1} , \tag{19}$$

where j denotes the j^{th} time step. This is a first-order prediction of $\mathbf{i}_{port}^{j+1}$ by the extrapolation of its prior two points. The superscript $(\cdot)$ means the iteration count of the NR method. We use the extrapolation, $\mathbf{i}_{port}^{j+1(0)}$ in (19), as the start point of the NR iterations.

For each NR iteration, Equation (16) has to be solved implicitly (LU for the whole matrix) once. However, only matrix $\mathbf{A}_n$ changes during each iteration; the rest of the matrix remains the same. Since the targets of our simulation are VLSI circuits with strong parasitic coupling, the linear circuit would be complex and makes its macro-model dense. To solve the whole matrix including linear part every NR iteration wastes time. Therefore, instead of solving the whole system equation (16) implicitly, we split it into two phases and solve equations (17) and (18) explicitly. For each Newton-Raphson iteration, k, we perform the explicit predictor:

$$\mathbf{A}_l \mathbf{x}_l^{j+1(k+1)^{(\frac{1}{2})}} = \mathbf{b}_l^{j+1} - \mathbf{E}_l^T \mathbf{i}_{port}^{j+1(k)} \tag{20}$$

$$\begin{bmatrix} \mathbf{A}_n & \mathbf{E}_n^T \\ \mathbf{E}_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_n^{j+1(k+1)^{(1)}} \\ \mathbf{i}_{port}^{j+1(k+1)^{(1)}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_n^{j+1(k+1)} \\ -\mathbf{E}_l \mathbf{x}_l^{j+1(k)^{(\frac{1}{2})}} \end{bmatrix} \tag{21}$$

In (20) and (21), $\mathbf{x}_l^{j+1(k+1)^{(\frac{1}{2})}}$ is an intermediate solution, and $\mathbf{x}_n^{j+1(k+1)^{(1)}}$ and $\mathbf{i}_{port}^{j+1(k+1)^{(1)}}$ are the prediction for the NR iteration $(k+1)$ of the $(j+1)^{th}$ time step. The physical meaning of this method is illustrated in Figure 5. In phase 1, we treat the initial branch current running through the ports as independent current sources[1] attached to the linear sub-circuit, and solve this sub-circuit alone. In phase 2, independent voltage sources[2] are attached to the ports using the values calculated in phase 1, and the linear companion model for the nonlinear circuit is generated and solved. Since the macro-model for linear circuit is fixed, we only have to perform LU decomposition to the nonlinear part for each NR iteration.

[1] or voltage sources if some elements in $\mathbf{i}_{port}$ represent voltage variables. This happens when the port connection is a mutual coupling between two sub-circuits or a control source. Please see Figure 2.

[2] or current sources for the similar reason.

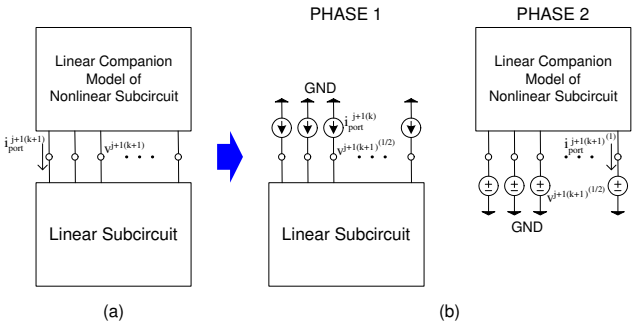

Figure 5: Physical meaning of the proposed partitioned explicit method (a) the original system (b) two phases of the method

4.2 Iterative Corrector

Obviously, if we use $\mathbf{i}_{port}^{j+1(k+1)^{(1)}}$ as the start point for next NR iteration, the error caused by the explicit predictor will reduce the converging rate of the NR. Thus, we propose an iterative corrector to improve the solution with only a small cost. Similar to (20) and (21), we iteratively solve:

$$\mathbf{A}_l \mathbf{x}_l^{j+1(k+1)^{(m+\frac{1}{2})}} = \mathbf{b}_l^{j+1} - \mathbf{E}_l^T \mathbf{i}_{port}^{j+1(k)^{(m)}} \tag{22}$$

$$\begin{bmatrix} \mathbf{A}_n & \mathbf{E}_n^T \\ \mathbf{E}_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_n^{j+1(k+1)^{(m+1)}} \\ \mathbf{i}_{port}^{j+1(k+1)^{(m+1)}} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_n^{j+1(k+1)} \\ -\mathbf{E}_l \mathbf{x}_l^{j+1(k)^{(m+\frac{1}{2})}} \end{bmatrix} \tag{23}$$

where (m) is the iteration count for the iterative corrector. If this procedure converges successfully, the final solution will approach to the original solution, which solves (16), of $(k+1)^{th}$ NR iteration. The method we proposed thus has the same accuracy with the ordinary simulation method. We will discuss the convergence property in the next subsection. The iterative corrector only has to perform forward/backward substitution. Thus the time saved in the predictor (do LU to smaller matrix) should dominate the time spend in the corrector (need more iterations) as long as the converging rate of the corrector is fast.

4.3 Convergence Analysis

In order to simplify the symbol used in this discussion, we ignore the superscripts j and k in (22) and (23). From these two equations, we have

$$\begin{bmatrix} \mathbf{x}_n^{(m+1)} \\ \mathbf{i}_{port}^{(m+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_n & \mathbf{E}_n^T \\ \mathbf{E}_n & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{b}_n \\ -\mathbf{E}_l \mathbf{A}_l^{-1} \left(\mathbf{b}_l - \mathbf{E}_l^T \mathbf{i}_{port}^{(m)} \right) \end{bmatrix} \tag{24}$$

$\mathbf{E}$'s are the adjacency matrices that contain one 1(or -1) in each row. Without losing generality, we can let $\mathbf{E}_n$ and $\mathbf{E}_l$ identity matrices, $\mathbf{I}$, by reordering the matrices $\mathbf{A}_n$ and $\mathbf{A}_l$. Hence, Equation (24)

$$\begin{aligned} &= \begin{bmatrix} \mathbf{A}_n & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{b}_n \\ -\mathbf{I}\mathbf{A}_l^{-1} \left(\mathbf{b}_l - \mathbf{I}\mathbf{i}_{port}^{(m)} \right) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & -\mathbf{A}_n \end{bmatrix} \begin{bmatrix} \mathbf{b}_n \\ -\mathbf{A}_l^{-1}\mathbf{b}_l + \mathbf{A}_l^{-1}\mathbf{i}_{port}^{(m)} \end{bmatrix} , \end{aligned} \tag{25}$$

and

$$\mathbf{i}_{port}^{(m+1)} = \mathbf{b}_n + \mathbf{A}_n\mathbf{A}_l^{-1}\mathbf{b}_l - \mathbf{A}_n\mathbf{A}_l^{-1}\mathbf{i}_{port}^{(m)} . \tag{26}$$

From the third term in the right side of (26), the iterative corrector converges if

$$|| - \mathbf{A}_n \mathbf{A}_l^{-1}|| < 1 \ . \tag{27}$$

A is the conductance matrix that represents the port characteristic of a sub-circuit. In our application, $\mathbf{A}_l$ is corresponding to the sub-circuit for interconnect parasitic, which is highly conductive (i.e. $||\mathbf{A}_l^{-1}|| < 1$). Usually the input impedance of a nonlinear circuit is large. Only during the gate transition time, both NMOS and PMOS are conducted and the impedance of the power-supply port becomes small. This means $\mathbf{A}_n$ is usually highly resistive (i.e. $||\mathbf{A}_n|| < 1$). Therefore, if $||-\mathbf{A}_n\mathbf{A}_l^{-1}|| << 1$, the sequence of the iterative corrector will be with a high converging rate.

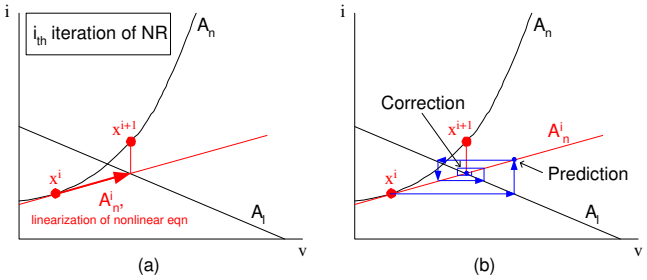

Figure 6: (a) The i^{th} iteration of NR; its solution is implicitly solved by $(A_n^i + A_l)$. (b) By solving A_n^i and A_l separately, the iterative predictor and corrector can converge to the same solution.

Figure 6 illustrates the convergence of the iterative predictor and corrector. Figure 6(a) shows the solution of the i^{th} NR iteration. As defined in this section, A_n and A_l are the matrices of nonlinear and linear sub-circuits respectively. A_n^i means the linearization of A_n at i^{th} iteration. The original NR method directly solve the whole matrix $(A_n^i + A_l)$, which is pretty time consuming.

Figure 6(b) shows the solution sequence of the iterative predictor and corrector. By explicitly solving A_n^i and A_l, this sequence converges to the same solution as that in (a). From this figure, the condition that results in fast convergence is a steep A_l and a flat A_n^i. This observation is the same as Equation (27).

If the sequence diverges in two successive iterations, we discard the result obtained from the predictor and implicitly solve Equation 16 for that NR iteration. The overhead of this divergence is performing the predictor and corrector once, whose cost is much smaller than implicitly solving the matrix. In our experiment, we never met divergence. All of the cases are solved by the explicit method.

5. SIMULATION RESULTS

We implemented the proposed hierarchical analysis, the partitioned explicit method, and the flat MNA simulator in C/C++ programming language. In order to have fair comparison, both methods use the same state-of-art sparse matrix solver. We also compared these methods with SPICE3[9]. The simulations are run on an Intel Pentium IV 1.4GHz system with RedHat 7.2 Linux operation system.

Figure 7 shows the waveforms of a clock tree simulation using SPICE3, our flat MNA, and the proposed hierarchical analysis. From this figure, the flat and hierarchical versions are equivalent. Our HiSIM simulator terminates the NR iteration when reaching either of the following conditions. 1.The nodal voltages converge to within a tolerance of $1\mu V$. 2.The nonlinear branch currents converge to within a tolerance of $1pA$. This meets the same accuracy requirement as SPICE3. The slight variation between our developed simulator and SPICE3 comes from the different choices of time steps. It is known that the NR algorithm fails to converge to a solution while the system is ill-conditioned. In order to guarantee reliability, techniques such as iteration damping and G_{min}-stepping that are adopted by SPICE3 are also implemented in HiSIM.

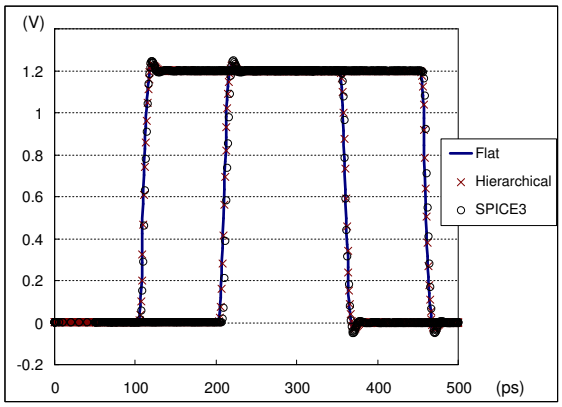

Figure 7: Waveforms of the flat MNA simulation, the hierarchical analysis, and SPICE3

Table 1 shows the runtime information of SPICE3, the flat MNA, and the hierarchical analysis. We tested various kinds of circuits. The first one is a clock tree with RLC interconnect model, which is used in Figure 7. The second circuit tested combines the first clock tree and a power-grid model to perform a clock and power-grid co-analysis. The following three cases are three bus structures. Each bit of them is driven by a drive buffer and ended with some load transistors. The buses are modeled with RLC PEEC model such that the inductance matrices are dense and are very difficult to analyze. The second column shows the number of nodes in the test cases. The third shows the number of linear and nonlinear devices respectively. Since the objective of HiSIM is interconnect-centric simulation, the cases we used all contain huge amount of linear elements while some nonlinear devices are present.

Besides runtime information, we also list the number of LU decompositions used to perform the simulation. The 6^{th} column shows the number of sub-circuits without and with nonlinear devices. A linear sub-circuit means a sub-circuit not containing nonlinear elements. We only have to factorize linear sub-circuit once and use the macro-model for later simulation. Therefore the average number of LU's for each NR iteration should be equal or less than the number of nonlinear sub-circuits. We count the number of LU's in DC solution and transient analysis separately. Divided by the total number of LU iterations, we obtain the average number

Circuit	# of nodes	# of elem. linear / nonlinear	SPICE3 runtime	Flat runtime	Hierarchical # of subckts linaer / nonlinear	DC # of LU NR iter	DC # of LU per NR	Transient # of LU NR iter	Transient # of LU per NR	runtime
CLK tree		45,812	2 min	2 min	6,169	10,540		6,778		1 min
w/o P/G	30,673	7,680	37 sec	26 sec	385	42	250.95	370	18.32	46 sec
CLK tree		148,856	>3	26 hour	6,170	24,217		62,038		58 min
w P/G	61,331	7,680	days	23 min	385	66	366.92	374	165.88	41 sec
8-bit		75,003	41 min	19 min	1	207		3,609		
bus	1,234	256	24 sec	37 sec	9	23	9.0	439	8.22	39 sec
16-bit		297,467	6 hour	2 hour	1	391		6,506		4 min
bus	2,466	512	36 min	24 min	17	23	17.0	471	13.81	39 sec
32-bit		1,482,230	>3	20 hour	1	759		15,760		42 min
bus	4,930	1,024	days	44 min	33	23	33.0	639	24.66	40 sec

Table 1: Runtime comparison of SPICE3, the flat MNA simulation, and the hierarchical analysis

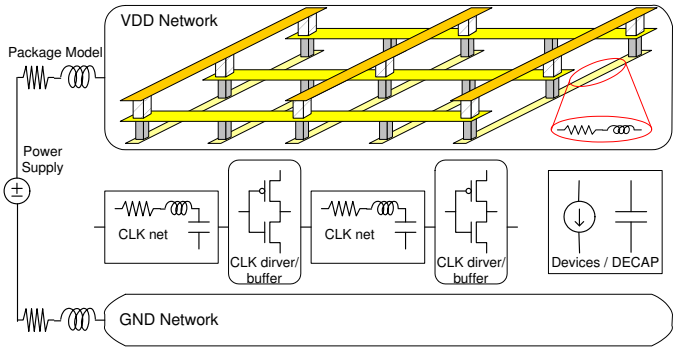

Figure 8: An example design and modeling of power-grid and clock net co-simulation, in which clock buffers are modeled by non-linear devices.

of LU's per NR iteration in columns 8 and 10. Obviously, these numbers are equal or less than the number of nonlinear sub-circuits, which demonstrates the temporary quiet described in Figure 4. The runtime result shows that our flat MNA is faster than SPICE3 while the circuit is large. Even more, the macro-modeling and hierarchical analysis technique improves the performance of HiSIM significantly. For the last test case, the hierarchical analysis gains 29.5x speed over the flat MNA.

To evaluate the performance of the proposed partitioned explicit method, there are two major concerns. First, the test cases must contain large amount of linear elements and some nonlinear devices. Second, the convergence condition is shown in Equation (27). If the partitioned explicit method does not converge, one implicit LU solution has to be executed. In this case, the cost of that NR iteration is the same as the hierarchical analysis but with a small overhead. Satisfying these two conditions, we thus use clock and power-grid co-analysis as test cases.

The power-grid and clock tree models used are illustrated in Figure 8. We use RLC interconnect model and transistor buffers to model the clock tree while the power grid contains only RLC linear elements. The current drained from other functional blocks are represented by independent current sources, which can be extracted by SPICE simulation with ideal power supply or estimated by some current estimation algorithms. Since clock timing is very sensitive to the power-delivery fluctuation, static analysis may not be accurate enough to guarantee the functionality of the system. The experimental setup shown in Figure 8 is hence necessary.

Table 2 shows the runtime information of the flat MNA, the hierarchical analysis, and the partitioned explicit method. The structure of this table is similar to that of Table 1. **# of iter** in columns 6-9 means the number of explicit corrector iterations. HiSIM can obtain the solution of each NR iteration by the explicit iteration solver as well as the implicit direct method. It only needs a few explicit iterations (about 2-3) to approach the original solution. From the runtime information, we also show that decomposing a small matrix and running a few explicit iterations are much faster than decomposing a large matrix. When the circuit size becomes larger, the runtime improvement is more significant.

6. CONCLUSION

With the diminishing timing and noise margin budgeting, the errors induced by static approximation may no longer be tolerable for modern VLSI design. Hence, in this paper we proposed an interconnect-centric circuit simulation method, which can handle nonlinear circuit simulation efficiently. Taking advantage of the design hierarchy, we proposed a decomposition-based macro-modeling technique, and a hierarchical framework to simulate large-scale nonlinear circuits with strong parasitic coupling. The simulation result shows that our hierarchical analysis method has dramatic speedup over the traditional flat MNA simulation method. In addition, we presented a partitioned explicit method to further improve our hierarchical simulation scheme. The partitioned explicit method partitions circuits into linear and nonlinear parts, and uses an iterative predictor and corrector to reduce the cost of directly solving huge matrices in each NR iteration. Excellent performance has been shown when performing this method to a clock-tree and power-grid co-analysis.

7. ACKNOWLEDGEMENT

This work was partially funded by National Science Foundation under grants CCR-0093309, CCR-0204468, and National Science Council of Taiwan, R.O.C. under grant NSC 92-2218-E-002-030. We greatly thank Dr. Noel Menezes for his valuable comments.

Circuit	# of nodes	# of elem. linear / nonlinear	Flat runtime	Hierarchical runtime	Partitioned Explicit DC # of iter / # of NR	Partitioned Explicit DC # of iter per NR	Partitioned Explicit Transient # of iter / # of NR	Partitioned Explicit Transient # of iter per NR	Partitioned Explicit runtime
clock tree w/		7,849	11 min		75		978		
2-layer power/ground	4,552	14	44 sec	4.3 sec	28	2.68	387	2.52	4.1 sec
clock tree w/		69,048	1 hour	10 min	132		993		1 min
4-layer power/ground	24,920	7,680	3 min	48 sec	45	2.93	424	2.34	44 sec
clock tree w/		125,039	20 hour	54 min	109		891		6 min
6-layer power/ground	37,983	7,680	11 min	28 sec	39	2.79	398	2.23	44 sec

Table 2: Runtime comparison of the flat simulation, the hierarchical analysis, and the proposed partitioned explicit method

8. REFERENCES

[1] R. Saleh, D. Overhauser, and S. Taylor. Full-chip verification of UDSM designs. In *International Conference on Computer-Aided Design*, pages 453–460, Nov 1998.

[2] M. Zhao, R. V. Panda, S. S. Sapatnekar, and D. T. Blaauw. Hierarchical analysis of power distribution networks. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 21(2):159–168, Feb 2002.

[3] J.N. Kozhaya, S.R. Nassif, and F.N. Najm. A multigrid-like technique for power grid analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(10):1148–1160, Oct 2002.

[4] R. Panda, D. Blaauw, R. Chaudhry, V. Zolotov, B. Young, and R. Ramaraju. Model and analysis for combined package and on-chip power grid simulation. In *International Symposium on Low Power Electronics and Design*, pages 179–184, Jul 2000.

[5] R. Saleh, S.Z. Hussain, S. Rochel, and D. Overhauser. Clock skew verification in the presence of ir-drop in the power distribution network. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(6):635–644, 2000.

[6] Shen Lin and Norman Chang. Challenges in power-ground integrity. In *International Conference on Computer-Aided Design*, pages 651–654, Nov 2001.

[7] Bhavana Thudi and David Blaauw. Non-iterative switching window computation for delay-noise. In *Design Automation Conference*, pages 390–395, Jun 2003.

[8] S. C. Chan, K. L. Shepard, and Dae-Jin Kim. Static noise analysis for digital integrated circuits in partially depleted silicon-on-insulator technology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(8):916–927, Aug 2002.

[9] L.W. Nagel. SPICE2: A computer program to simulate semiconductor circuits. *U.C. Berkely*, ERL Memo ERL-M520, 1975.

[10] Delivering a full-chip hierarchical circuit simulation & analysis solution for nanometer designs. Technical report, Nassda Corporation.

[11] Zhao Li and C.-J. Richard Shi. SILCA: Fast-yet-accurate time-domain simulation of VLSI circuits with strong parasitic coupling effects. In *International Conference on Computer-Aided Design*, pages 793–799, Nov 2003.

[12] T. L. Pillage, R. A. Rohrer, and C. Visweswariah. *Electronic Circuit & System Simulation Methods*. McGRAW-Hill, 1994.

[13] V. Litovski and M. Zwolinski. *VLSI Circuit Simulation and Optimization*. Chapman & Hall, 1997.

[14] W. C. Ho, D. A. Zein, A. E. Ruehli, and P. A. Brennan. An algorithm for DC solution in an experimental general purpose interactive circuit design program. *IEEE Trans. on Circuits and Systems*, CAS-24(8):416–421, 1977.

[15] T. N. Najibi. Continuation method as applied to circuit simulation. *IEEE Citcuits and Devices Magazine*, 5(5):48–49, 1989.

[16] A.E. Ruehli. Inductance calculations in a complex integrated curcuit environment. *IBM Journal of Research and Development*, pages 470–481, Sep 1972.

[17] D. Peaceman and J. H.H. Rachford. The numerical solution of parabolic and elliptic differential equations. *J. Soc. Indust. Appl Math*, 3:p28–41, 1955.

[18] J. K. White, A. Sangiovanni-Vincentelli, A. E. Ruehli, and F. Odeh. Waveform relaxation: Theory and practice. *Trans. Soc. Computer Simulation*, 2:95–133, Jun 1985.