

VariaSim: Simulating Circuits and Systems in the Presence of Process Variability

Technical Report #2007-3, June 2007

Bogdan F. Romanescu, Michael E. Bauer, Sule Ozev, and Daniel J. Sorin

variasim@ee.duke.edu

Department of Electrical and Computer Engineering

Duke University

<http://www.ee.duke.edu/variasim>

Abstract

In this paper, we present VariaSim, the publicly available Static Statistical Timing Analysis (SSTA) Tool from Duke University. VariaSim enables researchers to analyze the impact of CMOS process variability on the behavior of circuits and systems.

1 Introduction

To successfully design circuits and systems that tolerate variability, we must understand and quantify the impact of variability on the behavior of real circuits. The traditional approach has been “corner analysis”, in which the worst-case delay (e.g., mean delay plus three times the standard deviation) is assumed for *every* gate along every circuit path. Corner analysis is extremely pessimistic and provides an upper bound on delay that is thus extremely loose (30-40% high in our experiments). Corner analysis is analogous to assuming that, on a very long car ride, you will be stopped at every single traffic light for the maximum possible time at each.

Instead of corner analysis, for very small circuits (e.g., less than a few thousand logic gates), we can use detailed circuit simulation with Monte Carlo selection of parameter values. That is, we simulate the circuit many times. In each simulation, we use the Monte Carlo method to choose the values of each of the four low-level parameters that we are considering to exhibit variability. The values for each parameter are chosen randomly from the probability distribution of possible values for that parameter. By analyzing the results across all of the simulations, we can discover the probability distribution of delays through the circuit. However, Monte Carlo analysis has a runtime that is polynomial in the number of gates, and it thus does not scale to non-trivial circuits.

To address this challenge, we have developed VariaSim, a statistical timing analysis tool that is an exten-

sion of previously developed models [1, 2, 3, 5, 6, 7]. VariaSim’s inputs are the circuit’s netlist and the expected process variability (e.g., the standard deviation of transistor gate length). Given these inputs, VariaSim computes the mean and standard deviation (σ) of the delay through each circuit path. This standard deviation reflects the overall path delay variability that is a result of the low-level process variability. We analytically combine the results from the most critical paths to determine the results for a complete circuit.

We have distributed VariaSim publicly, so that the research community can use it to:

- Study impact of process variability on circuits and systems.
- Develop circuit and systems that can tolerate process variability.

In the rest of this paper, we describe our model (Section 2) and how to use the publicly available distribution of the model (Section 3).

2 VariaSim: An SSTA Tool for Determining the Impact of Variability

Because no current SSTA tool was publicly available, we have developed and distributed our own tool, VariaSim. VariaSim is most similar in approach to the SSTA tool proposed by Agarwal et al. [1].

2.1 High-Level Overview

A high-level overview of VariaSim is illustrated in Figure 1. VariaSim’s primary inputs are a description of a circuit and the means and variances¹ of the low-level process parameters (L , W , T_{ox} , V_{t0}). VariaSim’s outputs

1. Variance is often denoted by σ^2 . It is the square of the standard deviation (σ).

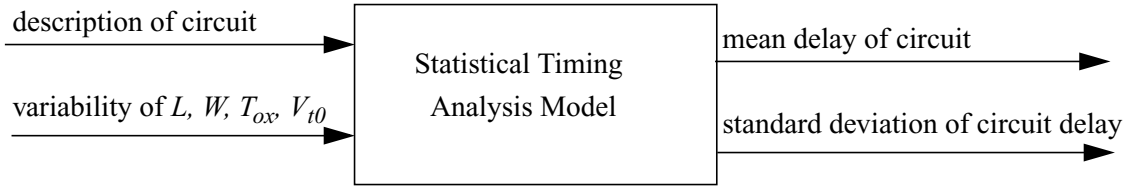


FIGURE 1. High-Level Overview of VariaSim

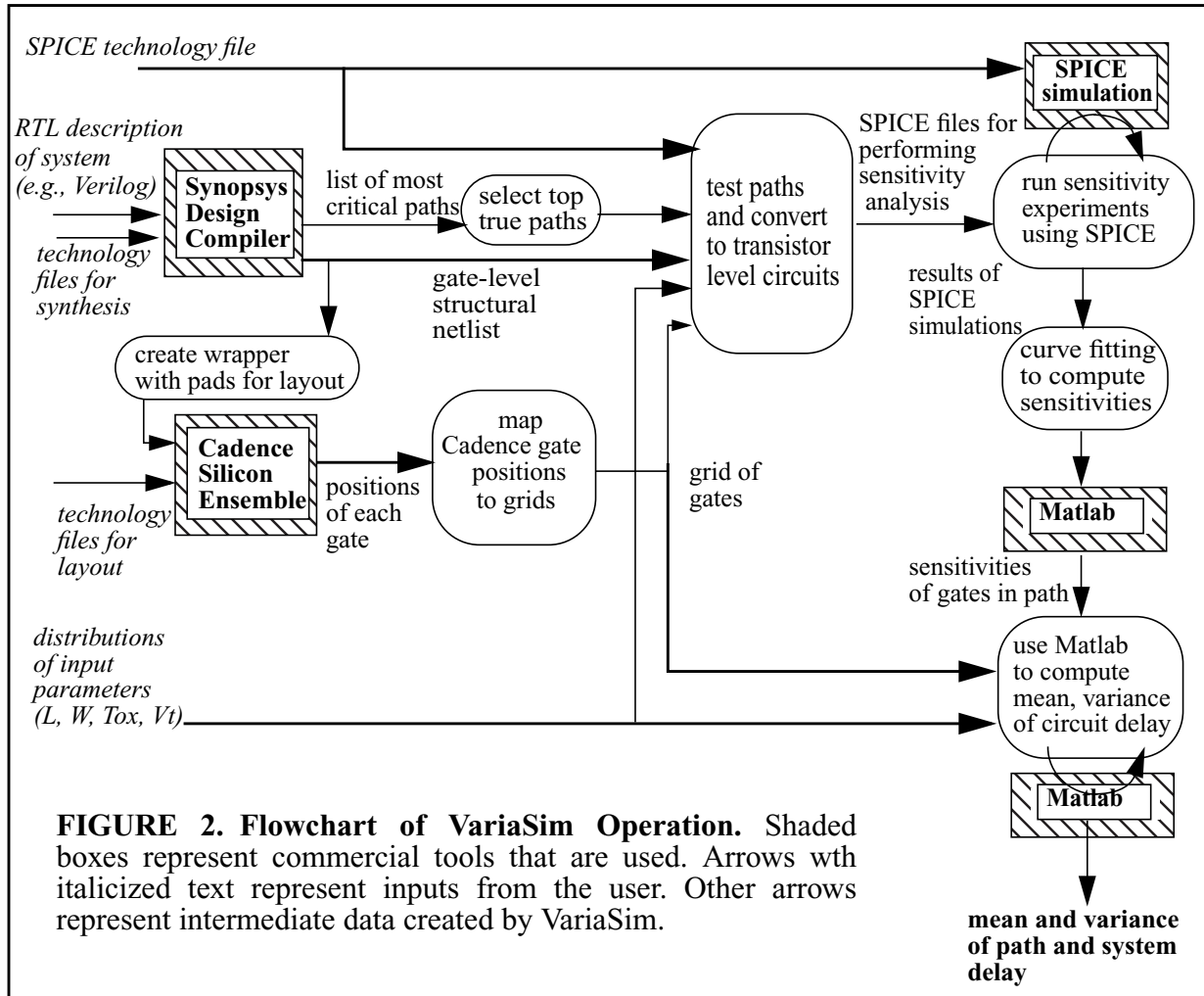


FIGURE 2. Flowchart of VariaSim Operation. Shaded boxes represent commercial tools that are used. Arrows with italicized text represent inputs from the user. Other arrows represent intermediate data created by VariaSim.

are the mean delay and variance of the delay through the circuit.

VariaSim can model spatial correlations similar to the work by Agarwal et al. [1]. For purposes of this discussion, we consider a single parameter, say L . A given chip has a chip-wide nominal value of L , L_{nom} , that is a baseline value that is shared by all transistors in the chip. We divide the chip area into four Level1 quadrants, each of which has different Level1 values for ΔL_1 . Each of these Level1 quadrants is then sub-divided into four Level2 quadrants, each of which has different Level2 values for ΔL_2 . Similarly, we have a Level3 and could, if

we wanted, have additional levels. The value of L for a transistor in a given Level3 quadrant is equal to the sum of L_{nom} and its Level1, Level2, and Level3 values for ΔL . This hierarchical model facilitates spatial correlations, because nearby transistors will often share Level1 and Level2 ΔL values. VariaSim chooses ΔL values at each level based on the spatial correlations that are input by the user.

For L , W , and T_{ox} , VariaSim allows the user to specify either spatial correlation (as described above) or completely random variation. For V_t , VariaSim allows only random variation. If the user specifies that all vari-

```

for each low-level parameter PARAM {
  pick 20 slightly different values of PARAM;
  for each value V {
    create SPICE file of circuit in which every transistor has PARAM=V;
    // these files used for die-to-die variability sensitivity analysis
  }
  for each switchable path {
    for each gate G on path {
      pick 10 slightly different values of PARAM;
      create SPICE file of circuit where every transistor in G has PARAM=V;
      // these files used for within-die variability sensitivity analysis
    }
  }
}

```

FIGURE 3. Creating SPICE Files for Sensitivity Analysis

ability is random (e.g., for L), then all ΔL values are zero and $L = L_{nom} + L_{rand}$.

2.2 Structure and Operation

Figure 2 presents the structure and operation of VariaSim. VariaSim consists of a handful of scripts that parse inputs and outputs, process and analyze data, and invoke several commercial CAD tools. VariaSim uses commercial CAD tools for various purposes, rather than reinventing them. It thus requires the user to have access to these tools, which are commonly used in both industry and academia. The commercial tools include Synopsys Design Compiler (for converting Verilog code to gate-level structural netlists), Cadence Silicon Ensemble (for determining the relative physical positions of gates in the structural netlists), SPICE (for circuit simulations), and Matlab (for data analysis).

VariaSim's first step is to take a Verilog description of the circuit and use Synopsys Design Compiler (DC) to process it. DC creates a gate-level structural netlist as well as an ordered list of the most critical paths in the circuit. DC determines this list by looking at how many gates are on each path, but it does not provide the inputs necessary to make them switch.

The outputs from DC are used in two different ways.

Mapping gates to quadrants. VariaSim passes the structural gate-level netlist to Cadence Silicon Ensemble, which labels the gates with their relative physical positions in the circuit. VariaSim processes this output from Silicon Ensemble in order to create a grid onto which it can place each gate. The smallest units of this grid correspond to Level3 quadrants in the spatial correlation model.

Performing sensitivity analyses. There are several steps in this process. First, VariaSim truncates the list of most critical paths, because only a certain percentage of paths can possibly affect the timing of the circuit, even in the presence of significant process variability. Second, for each path in this truncated list, VariaSim tests the circuit to find a set of inputs to make the path switch. If the path cannot be made to switch despite VariaSim's best efforts (this is an NP-hard problem), the path is discarded from the analysis. If the path can be made to switch, VariaSim produces multiple transistor-level netlists of the entire circuit. This process is best illustrated with pseudo-code, as shown in Figure 3. Third, VariaSim simulates each of these circuit netlists with SPICE, and VariaSim parses the results to determine the sensitivity of circuit performance to the low-level parameters.

Once VariaSim has generated the grid of gates and the results of the sensitivity analysis, it uses Matlab to compute the mean and standard deviation of the delay of each path. VariaSim then uses a previously validated approximation [4] to determine the mean and standard deviation of the delay of the entire circuit.

3 Using the SSTA Tool

VariaSim can be downloaded at <http://www.ee.duke.edu/variasim/>

This website provides the software as well as documentation for using the software. VariaSim requires the user to have access to a Verilog synthesis tool, a layout tool, and Matlab. The downloadable version of VariaSim assumes that the synthesis tool is Synopsys Design Compiler and that the layout tool is Cadence Silicon

Ensemble. Using other tools will require some modification to VariaSim's scripts.

Send comments and questions to variasim@ee.duke.edu.

4 References

- [1] A. Agarwal, D. Blaauw, and V. Zolotov. Statistical Timing Analysis for Intra-Die Process Variations with Spatial Correlations. In *Proceedings of IEEE ICCAD*, pages 900–907, Nov. 2003.
- [2] C. Amin et al. Statistical Static Timing Analysis: How Simple Can We Get? In *Proceedings of the 42nd Design Automation Conference*, pages 652–657, June 2005.
- [3] H. Chang and S. S. Sapatnekar. Statistical Timing Analysis Considering Spatial Correlations Using a Single Pert-like Traversal. In *Proceedings of International Conference on Computer Aided Design*, pages 621–625, Nov. 2003.
- [4] C. E. Clark. The Greatest of a Finite Set of Random Variables. *Operations Research*, 9(2):145–162, Mar.-Apr. 1961.
- [5] C. Visweswariah et al. First-Order Incremental Block-Based Statistical Timing Analysis. In *Proceedings of the 41st Design Automation Conference*, pages 331–336, June 2004.
- [6] Y. Zhan, A. J. Strojwas, X. Li, and L. T. Pileggi. Correlation-Aware Statistical Timing Analysis with Non-Gaussian Delay Distributions. In *Proceedings of the 42nd Design Automation Conference*, pages 77–82, June 2005.
- [7] L. Zhang et al. Correlation-Preserved Non-Gaussian Statistical Timing Analysis with Quadratic Timing Model. In *Proceedings of the 42nd Design Automation Conference*, pages 83–88, June 2005.