

# Transistor Level Gate Modeling for Accurate and Fast Timing, Noise, and Power Analysis

S. Raja, F. Varadi, M. Becer, and J. Geada  
CLK Design Automation, Littleton, MA

## ABSTRACT

Current source based cell models are becoming a necessity for accurate timing and noise analysis at 65nm and below. Voltage waveform shapes are increasingly more difficult to represent as simple ramps due to highly resistive interconnects and Miller cap effects at receiver gates. Propagation of complex voltage waveforms, and accurate modeling of non-linear driver and receiver effects in crosstalk noise analysis require accurate cell models. A good cell model should be independent of input waveform and output load, should be easy to characterize and should not increase the complexity of a cell library with high-dimensional look-up tables. At the same time, it should provide high accuracy compared to SPICE for all analysis scenarios including multiple-input switching, and for all cell types and cell arcs, including those with high stacks. It should also be easily extendable for use in statistical STA and noise analysis, and one should be able to simulate it fast enough for practical use in multi-million gate designs. In this paper, we present a gate model built from fast transistor models (FXM) that has all the desired properties. Along with this model, we also present a multi-threaded timing traversal approach that allows one to take advantage of the high accuracy provided by the FXM, at traditional STA speeds. Results are presented using a fully extracted 65nm TSMC technology.

## Categories and Subject Descriptors

J.6 [Computer-Aided Engineering]: Computer-aided Design; I.6.5 [Simulation and Modeling]: Model Development

## General Terms

Algorithms

## Keywords

gate modeling, timing, crosstalk, statistical, multi threaded

## 1. INTRODUCTION

Traditional cell timing models provide delay and output slew information as a function of input slew and output load

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC 2008, June 8–13, 2008, Anaheim, California, USA.

Copyright 2008 ACM ACM 978-1-60558-115-6/08/0006 ...\$5.00.

for a given cell arc. To analyze cell and net arcs using these models, an iterative Thevenin model and  $C_{\text{eff}}$  procedure is employed [1]. In the presence of highly resistive interconnects with significant shielding or with high crosstalk impact, these linearized gate models lack accuracy.

Recognizing these issues, EDA vendors recently proposed more detailed cell timing models where, for each timing arc, multiple measurements from the output waveform are recorded [2, 3]. While these models provide better accuracy due to increased resolution of characterization information, they are still input slew- and output load-dependent, are not suitable for multi-input switching analysis, and require a  $C_{\text{eff}}$  computation at each time step of their application.

In [10], a multi-port current source model was proposed where each pin of a cell is modeled with multi-dimensional voltage dependent current and charge elements. The multi-port nature of the model allows accurate modeling of multi-input switching, while accurately representing all non-linear capacitance effects between all input/output pins of a cell. However, model table dimensions increase prohibitively with multi-input cells. The authors also reported decreased accuracy for high-stack cells.

The accuracy problem for high-stack cells, as reported in [10], results from the fact that these cell-based models do not capture the staggered change in the states of stacked transistors. For example, consider a 3-input Nand gate as shown in Figure 1, with a rising transition at pin A3. Initially, there is no current flow in the N-stack except for a small leakage current. When A3 rises, transistor M3 turns on and starts to discharge node N2. However, due to Miller capacitance between A3 and N2, transistor M2 is turned off and does not conduct current until node N2 is discharged below  $V_{dd} - V_t$ . Similarly, transistor M1 will start discharging node X when N1 is discharged enough to turn on M1.

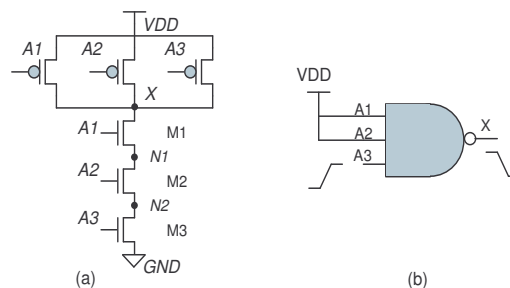


Figure 1: (a) Transistor circuit of 3-input Nand gate (b) Excitation for arc  $A3 \rightarrow X$

This causes a water-fall effect on the voltage waveforms on N2, N1, and X, and results in a delayed response on the output node X. When this A3→X arc is modeled in a cell based CSM, current would start flowing as soon as A3 rises, missing the delayed response at the output. A cell based CSM also cannot adequately model the capacitive effects on the stack nodes N2, N1, which leads to an inaccurate prediction of the slew rate at the output as well.

In static timing analysis (STA), which is one of the crucial steps in gate level design flow, delay modeling has always focused on models that represent gates as black boxes. All models mentioned above fall into this category. The motivation behind this abstraction has been performance and accuracy. With recent proposals of current source models, there is a clear trend to sacrifice some performance for increased accuracy. However, as mentioned above, even the most advanced of these cell based CSMs do not have universal high accuracy for all types of gates. This has also been recognized in [11] where the authors extend their work in [10] by exposing selected internal nodes as virtual ports during characterization, so that internal states of the cell can be represented in the model. Clearly, a modeling and analysis approach that is accurate within a few percent of SPICE, uniformly for all gate types and arcs, is required. This approach should also be fast enough for practical timing and noise analysis on multi-million gate designs.

One way to attack this problem is to model each transistor individually. One extreme way of doing this is simply running SPICE. However, such an approach would be dominated by transistor model (such as BSIM4 [4]) evaluations and would not be practical. Simplified modeling of MOS transistors for fast simulation has attracted significant interest in recent years. In this well studied field of computer aided design, it has been widely recognized that table models can provide both high speed and accuracy for MOS [15], and SOI [16] transistors. Such fast simulators have been used in transistor-level timing analyzers [5], or even selectively in gate-level analysis tools to increase accuracy when needed [8]. However, such fast SPICE simulators have not been proposed to be used as the main timing calculation engine of a gate level STA or noise tool to be run on multi-million gate designs.

Therefore, in this paper we propose a gate modeling and analysis approach which satisfies both accuracy and speed concerns mentioned above. In our model, we represent each transistor in a gate with a model (FXM) composed of a DC current source and a set of configuration-based linear capacitances. This is, in fact, a simplified version of fast MOS models proposed earlier. We show that this model is accurate enough for all required purposes.

The desired combination of accuracy and speed cannot be reached without a significant change in how a timing/noise analysis tool utilizes available compute resources. Today, compute power is cheap with multiple processors on a machine, multiple CPU cores in a single processor chip, and multiple computing threads available in each core. In addition to the FXM based gate modeling, we also propose a better utilization of available computing threads by the analysis tool which results in significant parallelization and large savings in run time. With these modeling and analysis approaches combined, SPICE level accuracy with traditional timing analysis speeds can be achieved.

The remainder of the paper is organized as follows. In

section 2, we explain the structure and characterization of proposed FXM. Model's usage in timing simulation is detailed in section 3. Section 4 presents the basics of our multi-threaded timing analysis tool. We demonstrate the accuracy of the proposed model on several test cases in section 5. Timing analysis speed using this model compared to traditional models is also demonstrated in this section. Section 6 contains the closing remarks.

## 2. A FAST TRANSISTOR MODEL (FXM)

The motivation of our proposal to use transistor-level cell models comes from the high accuracy of the transistor representation of cells. In this section we discuss accuracy versus performance tradeoffs and propose a representation that satisfies the accuracy and performance needs.

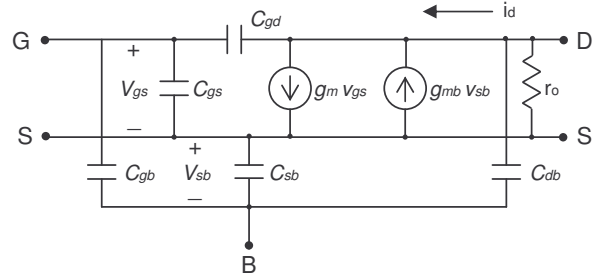


Figure 2: Small signal high frequency MOSFET model

A MOSFET high-frequency, small-signal model is shown in Figure 2. BSIM transistor models determine capacitances and currents by solving complex equations which are functions of the process parameters contained in the model. The proposed current source model for a MOSFET represents the drain current as a single current source that depends on terminal voltages. This current source can simply be represented as a pre-characterized 3-d lookup table depending on the gate, drain and source voltages. The simplified FXM representation of a MOSFET is shown in Figure 3.

### 2.1 Transistor Capacitances

Gate capacitances  $C_{gd}$ ,  $C_{gs}$  and  $C_{gb}$  are non-linear and depend on the operating region of the transistor.

The capacitances  $C_{sb}$  and  $C_{db}$  are the junction depletion capacitances, and are constant for all operating regions of the transistor.

The most accurate way of modeling the non-linear capacitances is to represent them as voltage dependent charge sources. Characterization of such a model would involve generating charge tables for a range of terminal voltages. The use of such a model would require the computation of  $dQ/dV_i$  at every time step where  $i$  is the gate, drain, and source terminal; resulting in a non-constant capacitance matrix. Although this approach would be most accurate compared to SPICE, performance would be a problem in an STA environment. In addition, characterization also becomes a runtime intensive task.

Using constant capacitances promises faster performance. One way to achieve this is to choose a single value for all the device capacitors. While this leads to relatively simple transistor models with sufficient accuracy for a large number

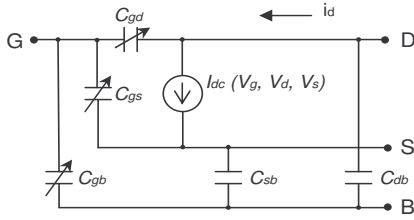


Figure 3: Current source model for a MOSFET

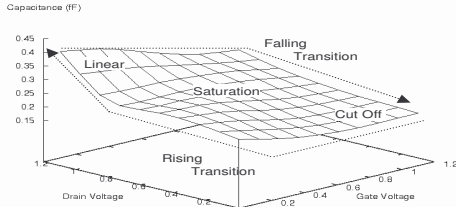


Figure 4:  $C_{gd}$  variation for a PMOS

of cases, there are cases when more elaborate models are needed.

Figure 4 shows the variation of  $C_{gd}$  capacitance for a PMOS with source held at supply voltage of 1.1v. It is clear that this significantly non-linear capacitance cannot be modeled accurately with a single capacitance as the transistor is used in different configurations.

In order to improve the accuracy while still maintaining the better performance of a constant capacitor model, we use capacitance values based on the initial operating state of the transistor. The justification behind this choice is the fact that the non-linear Miller capacitances  $C_{gd}$  and  $C_{gs}$  influence the output waveform mostly at the beginning of the input transition. For example, for a falling transition through a PMOS channel, gate capacitances characterized at *linear* state are used for the entire transition.

## 2.2 Statistical extension

In addition to the nominal values for the dc current source and configuration based linear capacitances, statistical extension of FXM also contains the sensitivity of these model elements to any statistical parameters of interest. These can be physical entities such as  $l_{eff}$ , or they can also be non-physical parameters derived from a principal component analysis.

## 2.3 Characterization

Standard cell libraries today consist of hundreds of cells. With many process corners, all previously proposed cell-based current source models require significant time to characterize. The proposed model has very simple characterization requirements that make characterization setup and runtime very fast. Since the proposed FXM is transistor-based, we need only characterize the unique transistors in the cell library, defined by the transistor's width, length, multiplication factor and other model parameters.

Characterization runs are performed using a SPICE tool such as Berkeley Spice3f5. DC IV curve of an FXM is char-

acterized by a DC sweep of the voltages at the gate, drain and source terminals. Transistor capacitances are characterized for two operating states of the transistor (*Cut-off* and *Linear*). Proper bias voltages are chosen for each state and the capacitances are characterized by applying a short voltage ramp on the terminals and measuring the incremental charge introduced.

Statistical extension of the model requires the computation of model elements above and below the nominal value (e.g.  $+1\sigma$  and  $-1\sigma$ ) of the statistical parameter being modeled, so that a numerical sensitivity can be computed. The exact nature of the characterization approach depends on the particular parameter being modeled.

Cell transistor topologies are also stored as part of a library characterization so that they can be reconstructed using FXM during analysis. A characterization program has been developed that reads standard cell SPICE sub-circuits, transistor models and characterization setup and generates the FXM based gate models in extended Synopsys Liberty format. On a TSMC 65nm standard cell library, we characterized 280 standard cells where FXM characterization was performed on 2867 unique transistors. Characterization runtime was 5 minutes on a 16-CPU box running at 1.8GHz. Statistical FXM characterization with 4 global and 4 local variables took 40 minutes.

## 3. MODEL USAGE IN TIMING SIMULATION

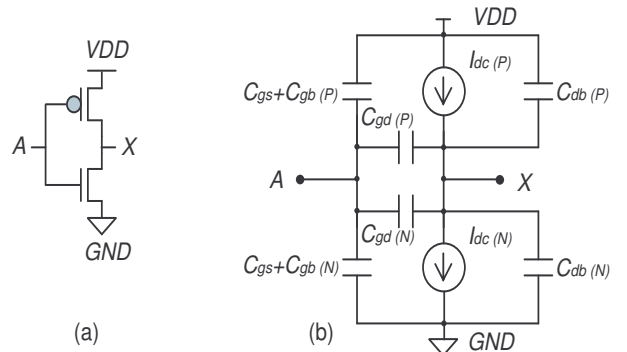


Figure 5: Transistor based CSM structure for an Inverter

During timing simulation of a gate or net arc, to account for the non-linearity of the driver and the receiver, an FXM based representation of the cells need to be constructed. From the transistor topology of the cell, each transistor is replaced by its equivalent FXM. Figure 5 shows an FXM representation of an inverter.

In extracted libraries, a schematic transistor is usually represented with multiple finger devices. There are also a significant number of parasitic R/C elements among these devices. This significantly increases the number of nodes and devices to be simulated. We employ several techniques to reduce the complexity of an FXM based cell representation with very little loss in accuracy. These techniques include reducing internal R/C nodes using transformations, and collapsing finger devices.

RC loads driven by the FXM based cells come from extraction tools in the form of a parasitic file, such as SPEF.

These linear RC networks are modeled by reduced order modeling techniques, such as PRIMA [12].

Note that the FXM based cell representations can be used for drivers of a net as well as the receivers.

#### 4. MULTI-THREADED TIMING ANALYSIS

The downside of using a more accurate timing model, such as the FXM described here, is that these approaches are typically significantly more compute intensive than the simpler methods they replace. In order to bring the runtime down to a practical realm, one approach is to distribute the work across multiple processors.

This can be done through explicit partitioning, which is typically implemented by running a pass over the data structures representing the design and partitioning the work into independent or semi-independent regions. Such approaches are problematic from several fronts:

- By requiring an extra computational pass, they limit the maximum possible improvement.
- It becomes increasingly hard to generate independent regions as the number of available processors is increased. Given that 16 CPU and more machines are increasingly available, this is a significant problem.
- They typically require complex locking at the boundaries of such regions and/or a further pass at the end to resolve all the data crossing between regions, again further limiting the benefits of threading.

Our objective is to maximize the performance improvement as additional processors are made available to the algorithm. To this end we developed a patent-pending, fine scale algorithm that requires no partitioning and that requires no significant locking or synchronized regions.

The fundamentals of the algorithm are as follows:

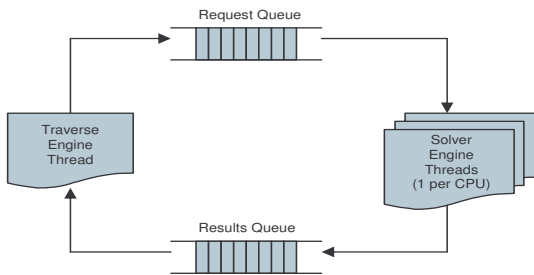


Figure 6: Threading Algorithm

The traversal engine thread is responsible for 2 tasks:

- Performing a breadth first traversal of the design and, as each cell has all its inputs made available, putting the request for the computation of the delay across those cells into the request queue.
- Once the set of all cells that have available inputs is exhausted, pulling the delay data from the queue and inserting the results into appropriate design data structures.

Effectively, this is the standard breadth first search (BFS) algorithm managed with an external stack, except that the external stack is separated (pushes are done into the request queue but pops occur from the results queue). Note that these queues are required to be thread safe and must safely permit multi-threaded simultaneous pushes/pops [7].

The traversal algorithm itself is insulated by this approach from having to deal with any threading issues or locking protocols; also the algorithm is completely neutral to the number of solver threads available and can easily be configured at runtime to run with any number of threads or even completely unthreaded. On the solver side, for maximum performance, one separate thread is created per available CPU. Each such thread can safely read any of the input data required for its calculations, as this algorithm guarantees that all such data is now stable and will not be further modified (and thus can be read lock free). The FXM (or any other calculation approach) can be performed safely with the only constraint being that any data generated be stored in a newly allocated data structure rather than being directly stored. This allows all the work in the solver to be performed without requiring any additional locking operations. Once the solve is completed, the result data structures are pushed back into the results queue and the thread continues to process the next available request.

The results have proven the viability of the approach, showing very good scaling across a range of CPUs:

Number of CPUs	Relative Performance
1	1.0
2	1.8
4	3.6
8	7.6
16	14.3

Table 1: Performance scaling data for complete solve

Relative performance numbers presented in Table 1 include setup costs of timing algorithms that cannot be threaded. This data has been collected from a number of large industrial designs ranging from a few million gates up to more than 9 million gates.

#### 5. RESULTS

Using the proposed FXM, we present voltage waveforms from several scenarios and compare them with transistor level SPICE simulations. The experiments were performed using a fully extracted TSMC 65nm library. Note that the time axis has been scaled in the figures.

Upper figure in Figure 7 shows the voltage waveform at the output of a 3-input nand gate for a timing arc from the bottom transistor on stack.

Lower figure in Figure 7 shows the accuracy of FXM when used in a multi-input switching scenario. The simulation consists of a 2-input NAND gate where both inputs are falling.

Upper figure in Figure 8, shows a waveform comparison between SPICE and FXMs under crosstalk conditions. We compare the waveforms when a noisy input signal is driving a gate which also has injected crosstalk noise at the output, as well as IR drop at its supply voltage.

All waveforms in above plots show a very good match

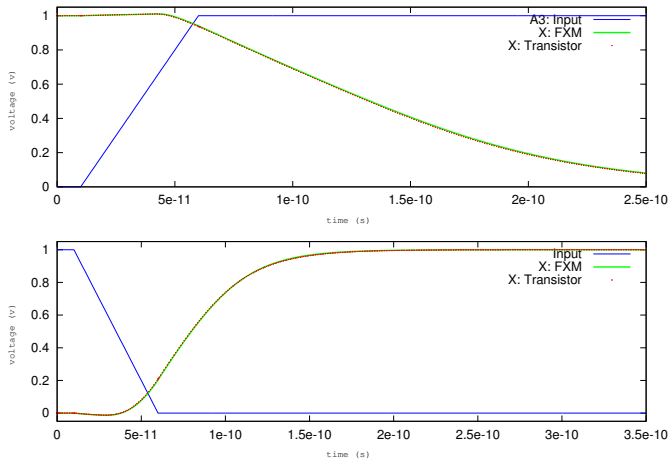


Figure 7: top: 3-input NAND gate, for arc  $A3 \rightarrow X$ , bottom: MIS for a 2-input NAND gate

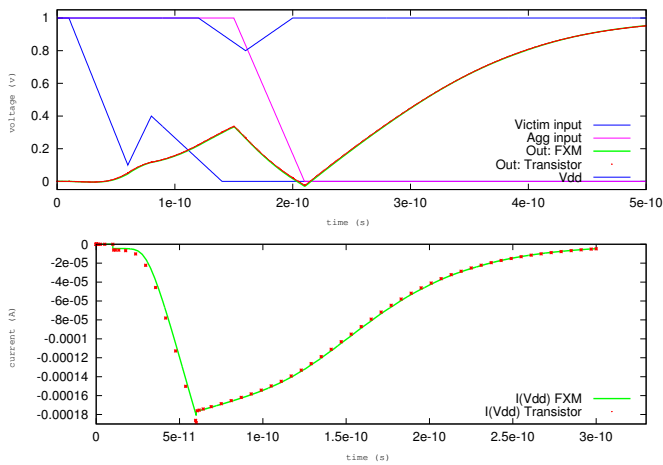


Figure 8: top: Victim delay noise in the presence of input noise and IR drop, bottom: Dynamic power

between transistor level SPICE simulation and FXM.

Lower figure in Figure 8, demonstrates another application of the FXM based cell models. As can be seen, current drawn from supply during a transition can be modeled very accurately using FXM. This results in the capability to perform dynamic power analysis using the same representation of a library.

In figure 9, we present a comparison of delay and slew measurements using our model and transistor level spice simulations for the entire TSMC 65nm library, for all possible gate arcs. Cells in this library are fully extracted with multiple finger devices and complex R/C networks among them. Note that this library includes gates with Miller capacitance effects to intermediate nodes, multi-input gates with high stack effects, Exclusive-OR gates using pass gates, etc. Using FXM based gate models demonstrate high accuracy for all type of gates, in all load and input slew conditions. Our analysis shows that we are within 5.7% of Spice for delay and slew. Note that 90% of cases are within 2%. Runtime comparison during this test showed that FXM simulation is faster than transistor level spice simulation by an average of

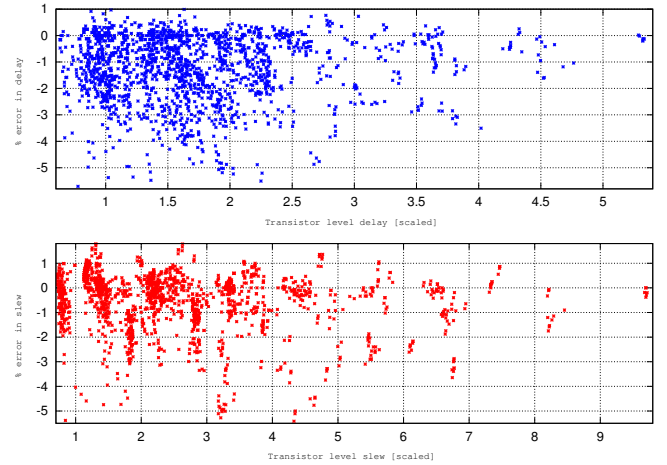


Figure 9: Delay and slew accuracy for fully extracted 65nm library

two orders of magnitude.

We also evaluated the statistical extension of FXM using the same library. FXM-based cell models enable the propagation of time-domain waveforms, without having to abstract them into delay/slew pairs. With statistical-FXM, in addition to the time-domain waveforms, simulation results in sensitivity waveforms that represent the sensitivity of the voltage waveform with respect to the statistical parameters being modeled. This representation can then be converted into a canonical arrival time and slew, and local variables can be collapsed into a single variable to prevent an explosion in the number of variables.

Figure 10 shows a sample waveform at the output of a statistical FXM-based gate model. Upper figure is the nominal voltage waveform. Lower figure shows several sensitivity waveforms, some of which come from the input waveform, some of which come from the statistical-FXM of this gate.

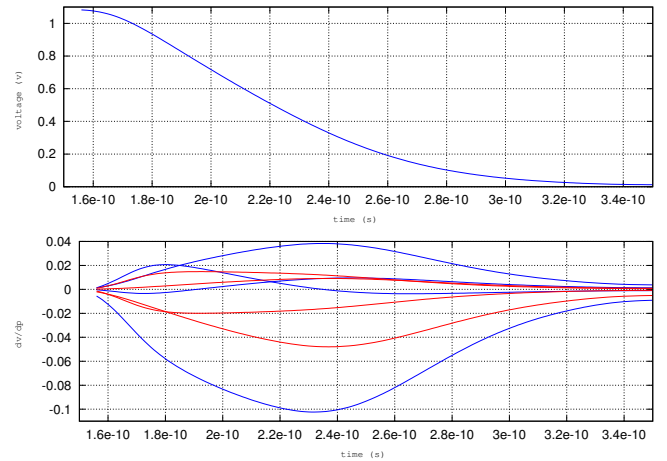
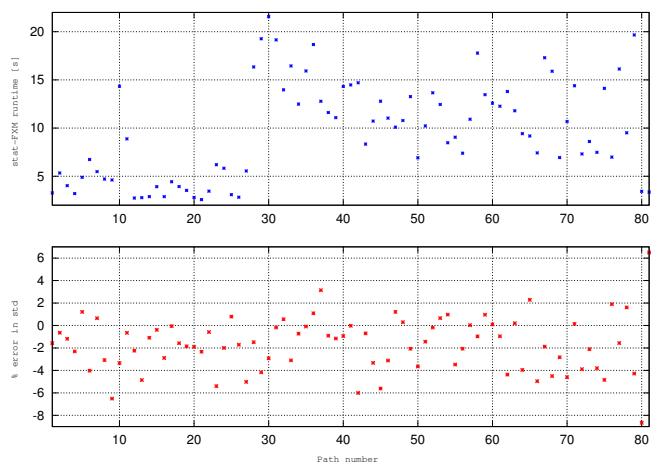


Figure 10: Simulation output using statistical-FXM

We tested the accuracy and runtime of statistical-FXM on a 65nm industrial design. After STA, Monte-Carlo analysis was performed on the most critical 81 paths using transistor level Spice simulations. Note that these simulations were done on the entire path, not on a stage by stage basis. In regards to the statistical parameters, we used 4 global

variables that are shared among all transistors, and 4 local variables of which each transistor had its own independent copy. We then analyzed these paths using statistical-FXM and compared the total standard deviation of arrival times at path end points with Monte-Carlo results. Average error in standard deviation was around 1%, with the worst outlier at 8.6%. In Figure 11, x-axis is the path number that we performed this statistical analysis. Upper figure shows runtimes where, for demonstration purposes, each path was run on a single thread. Lower figure shows percentage error of std with respect to transistor level Monte-Carlo. Runtime of this statistical-FXM based analysis for these 81 paths on an 8-CPU machine was 101 seconds.



**Figure 11: Statistical-FXM runtime and accuracy**

Finally, we compare runtime on several large industrial designs using delay modeling based on traditional delay / slew based libraries versus the proposed FXMs. Based on data collected from several large industrial designs, the runtime hit of using FXM based models is between 3x - 10x depending on the accuracy requirements. As evident by almost-linear scaling properties of our threading algorithm, STA runs using traditional libraries are easily sped up more than an order of magnitude with 16 processors. This is a significant advantage in a timing closure effort, enabling overnight runs to be completed in around an hour. On the other hand, FXM enables much desired accuracy in STA at practical runtimes.

## 6. CONCLUSION

We presented an accurate cell modeling technique based on a transistor-level representation (FXM). FXMs result in input waveform/output load-independent cell models, they speed up library characterization, and reduce library complexity. We presented the accuracy and speed of the model compared to transistor level SPICE simulations in several interesting scenarios. We showed the capabilities of the statistical extension to FXM. We also presented a multi-threaded timing traversal approach. We showed that with proper utilization of available compute resources, it becomes practical to use high accuracy cell models such as FXM, in multi-million gate STA runs.

## 7. REFERENCES

- [1] F. Dartu, N. Menezes, and L.T. Pileggi "Performance computation for precharacterized CMOS gates with RC loads," In *IEEE Transactions on CAD*, vol.15, no.5, pp.544–553, May 1996.
- [2] Composite Current Source, Synopsys 2005, "CCS timing white paper," In [http://www.synopsys.com/products/solutions/galaxy/ccs/cc\\_source.html](http://www.synopsys.com/products/solutions/galaxy/ccs/cc_source.html)
- [3] Cadence Technical Paper, 2005 "Delay calculation meets the nanometer era," In [http://www.cadence.com/products/digitalic/tech\\_info.aspx](http://www.cadence.com/products/digitalic/tech_info.aspx)
- [4] BSIM4 Home Page In <http://www-device.eecs.berkeley.edu/~bsim3/bsim4.html>
- [5] "PathMill: Transistor-level static timing analysis," In <http://www.synopsys.com/products/analysis/pathmill.ds.pdf>
- [6] J.F. Croix, and D.F. Wong "Blade and Razor: Cell and interconnect delay analysis using current-based models," In *Proceedings of DAC 2003*, pp.386–389, June 2003
- [7] E. Ladan-Mozes, and N. Shavit "An optimistic approach to lock-free FIFO queues," In *Proceedings of International Conference on Distributed Computing*, 2004
- [8] R. Levy, D. Blaauw, G. Braca, A. Dasgupta, A. Grinshpon, C. Oh, S. Sirichotiyakul, and V. Zolotov "Clarinet: A noise analysis tool for deep submicron design," In *Proceedings of DAC 2000*, pp.233–238, June 2000
- [9] I. Keller, K. Tseng, and N. Verghese "A robust cell-level crosstalk delay change analysis," In *Proceedings of ICCAD*, pp.147–154, November 2004
- [10] C. Amin, C. Kashyap, N. Menezes, K. Killpack, and E. Chiprout "A multi-port current source model for multiple-input switching effects in CMOS library cells," In *Proceedings of DAC 2006*, pp.247–252, Jun 2006
- [11] C. Kashyap, C. Amin, N. Menezes, and E. Chiprout "A nonlinear cell macromodel for digital applications," In *Proceedings of ICCAD 2007*, pp.678–685, Nov 2007
- [12] A. Odabasioglu, M. Celik, and L. T. Pileggi "PRIMA: Passive reduced-order interconnect macromodeling algorithm," In *Proceedings of ICCAD 1997*, pp.58–65, Nov 1997
- [13] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan "First-order incremental block-based statistical timing analysis," In *Proceedings of DAC 2004*, pp.331–336, Jun 2004
- [14] A. Agarwal, D. Blaauw, and V. Zolotov "Statistical timing analysis for intra-die process variations with spatial correlations," In *Proceedings of ICCAD 2003*, pp.900–907, Nov 2003
- [15] T. Shima, T. Sugawara, S. Moriyama, and H. Yamada "Three-dimensional table look-up MOSFET model for precise circuit simulation," In *IEEE Journal on Solid State Circuits*, vol.SC-17, no.3, June 1982
- [16] D. Nadezhin, et.al "SOI transistor model for fast transient simulation," In *Proceedings of ICCAD*, pp.120-127, November 2003