Considering Process Variations During System-Level Power Analysis

Saumya Chandra[†‡] Kanishka Lahiri[‡] Anand Raghunathan[‡] Sujit Dey[†]

[†]Dept. of ECE, University of California, San Diego (saumya,dey@ece.ucsd.edu)
[‡]NEC Laboratories America, Princeton, NJ (klahiri,anand@nec-labs.com)

ABSTRACT

Process variations will increasingly impact the operational characteristics of integrated circuits in nanoscale semiconductor technologies. Researchers have proposed various design techniques to address process variations at the mask, circuit, and logic levels. However, as the magnitude of process variations increases, their effects will need to be addressed earlier in the design cycle.

In this paper, we propose techniques for accurately and efficiently incorporating the effects of process variations into system-level power estimation tools. To motivate our work, we first study the impact of process variations on the power consumption of an example System-on-Chip (SoC). We consider simple extensions of current approaches to system-level power estimation (spreadsheet-based and simulation-based power estimation), and demonstrate their limitations in performing variation-aware power estimation. We propose a system-level power estimation methodology that can accurately and efficiently analyze the impact of process variations on SoC power. The proposed methodology combines efficient trace-based analysis, power-state based leakage modeling, and Monte Carlo sampling. The key benefit of the proposed methodology is that it captures the necessary inter-dependencies while avoiding iterative system-level simulation. Our implementation of the proposed techniques within an in-house system-level power estimation framework indicates 2-5 orders of magnitude efficiency gains, with negligible loss in accuracy, compared to direct Monte Carlo techniques that require iterative system simulation.

Categories and Subject Descriptors: C.5.4 [VLSI Systems]

General Terms: Algorithms, Design, Experimentation

Keywords: Process variations, Power analysis, Power Estimation, Low Power Design, System-on-Chip

1. INTRODUCTION

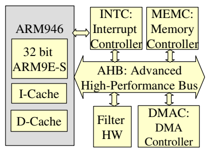
Integrated circuits (ICs) fabricated using nanoscale technologies are expected to be increasingly prone to manufacturing induced variations, which cause the characteristics of devices to vary both within a die and across dies [1]. These variations cause the performance and power consumption of circuits, and hence the systems that contain them, to display statistical, rather than deterministic behavior. Traditional design methodologies, based on typical and worst-case circuit models, break down in the face of increasing variations, resulting in over-design, increased design effort, decreased yield, or an inability to meet design goals.

Recognizing the above challenges, various techniques at different stages of the design flow are being researched to aid in IC design in the presence of variations. The efforts in this area have thus far focused on addressing the problem at the mask [2], circuit [1, 3], and logic levels [4, 5, 6]. While these techniques have shown promise (and are already being incorporated into commercial IC design flows), they cannot completely address the problem, especially in the face of continually increasing process variations. Variation-aware design has only recently started to receive attention at higher levels of the design flow, namely, at the architecture and system levels. Statistical models to incorporate the impact of variations on microprocessor performance and power consumption were proposed in [7]. Tradeoffs between throughput, power, and area in parallel architectures under process variations were studied in [8]. The effects of process variations on embedded SRAM memory architectures were studied in [9] and reconfigurable buffers were exploited to optimize power and performance under variations.

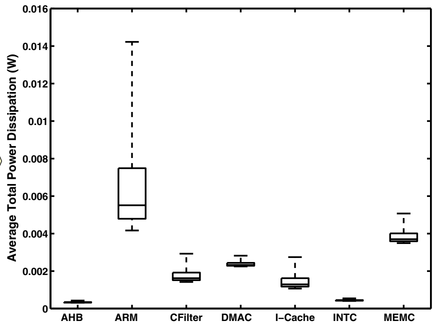
Key to designing variation-tolerant systems is the availability of accurate and efficient analysis tools that predict the impact of variations on design metrics, such as the system-level power consumption. However, this area has not received much attention. In this work, we propose an efficient methodology based on efficient trace analysis, power-state based leakage modeling, and Monte Carlo sampling to provide SoC designers with feedback about power consumption under process variations.

2. SOC POWER UNDER VARIATIONS

In this section, we consider the example SoC architecture illustrated in Figure 1(a). The SoC implements an image processing application that consists of software running on an ARM946 processor [10], and dedicated hardware (`Filter_HW`) that accelerates pixel-level filtering operations. In addition, the SoC contains the AHB on-chip bus [11], an integrated memory controller, and an interrupt controller. The SoC is implemented using a commercial 90 *nm* standard cell library [12] and operates at a frequency of 206 *Mhz* and a voltage of 1 *V*. An in-house, cycle-accurate, simulation-based system-level power analysis tool [13] was used to generate dynamic and leakage power traces for each component while executing a specific test-bench. We used Monte Carlo techniques (described in Section 3) to estimate the impact of variations in leakage power, due to chip-to-chip variations in effective channel length (L_{eff}). For this study, we assumed that L_{eff} follows a normal distribution with $\mu = 90$ *nm* and $3\sigma/\mu = 30\%$. The power variations thus estimated for each component are captured using a box-whisker representation as shown in Figure 1(b). The lower and upper extremities of each box represent the 25th and 75th percentiles of a component's average power consumption (including both dynamic and leakage power). The whiskers denote the minimum and maximum values. For example, for the ARM processor, the inter-quartile range (the box height) is 25% of its average power, suggesting that variations significantly impact its power characteristics. However, for the AHB it is only 8%. This difference is explained as follows. While leakage power is highly sensitive to channel length variations, dynamic power is relatively immune. Hence, components for which leakage accounts for a greater portion of their total power consumption display higher total power variations. The breakdown of a component's total power into leakage and dynamic power is determined by how much time it spends in its different power-states (active, idle, sleep, deep-sleep, *etc.*). In the example, the ARM processor spends a significant amount of time in the idle state waiting for the `Filter_HW` and therefore, is more affected by variations. The AHB is almost always active, serving requests from either the ARM processor or other HW. Its power consumption consists largely of dynamic power, and is hence less susceptible to variations. This example suggests that the extent to which variations affect individual component power characteristics depends critically on *component workload profiles* and *power-states*. Accounting for such inter-dependencies accurately and efficiently is a key objective of the proposed methodology.



(a)



(b)

Figure 1: Case study in chip-to-chip power variations for an example SoC: (a) system-level block diagram; (b) inter-quartile range of power variations for system components

3. CONSIDERING VARIATIONS IN POWER ANALYSIS

The goal of variation-aware power analysis is to generate a power *distribution* rather than a deterministic estimate. Moreover, this distribution must be estimated accurately and efficiently. There are two main approaches currently used for system-level power estimation: (i) enhancing system-level simulation with power models for various system components, and (ii) simple spreadsheet analysis based on rough metrics such as total gate count, switching activity factors, *etc*. We consider simple extensions of these approaches to consider variations, and evaluate their merits and drawbacks.

Direct Monte Carlo simulation: In this method, full-system simulation-based power estimation is iteratively performed in order to generate a power consumption distribution for the SoC. The power models for system components are initially constructed based on nominal (typical) values of process parameters. For each simulation, the power models are refined by randomly generating a sample point from a pre-defined distribution of process parameters (*e.g.,* transistor channel lengths), and calculating a variation factor for dynamic and leakage power. The power models are assumed to be sensitive to the power-states (active, idle, sleep, *etc.*) of the respective components, since the power variability will be different in different states. When simulation-based system-level power analysis is performed, functionality and power-state transitions are simulated in an integrated manner, and total power consumption is determined. This is repeated for a desired number of sample points in order to obtain a distribution of total power consumption for the SoC under process variations.

The larger the sample size, the higher is the degree of confidence in the accuracy of the computed power distribution. For the example SoC shown in Figure 1(a), we calculated that in order to be 95% confident that the estimated mean of the power distribution differs from the actual mean by no more than 5%, at least 607 simulations would be required, which is a computationally challenging task since each simulation may require anywhere from several minutes to hours, depending on the length of the simulation trace. In summary, direct Monte Carlo simulation can be accurate, but is too time consuming to use for architectural exploration.

Spreadsheet-based analysis: In this method, simple equations are used to analytically calculate the distribution of leakage power, given SoC characteristics such as the gate count and activity profile, and a distribution of process parameters. It is possible to analytically relate variations in transistor leakage to variations in process parameters such as channel length and oxide thickness [5, 6]. The SoC's characteristics, such as gate count and activity factor, are then used to estimate the average number of leaking transistors, and compute distributions for leakage power and total SoC power.

Figure 2 compares the distribution obtained through such spreadsheet-based analysis to one obtained via direct Monte Carlo simulations under a given test bench. Clearly, there is a notable discrepancy between the two distributions. This discrepancy arises because, for a specific workload, the spreadsheet-based approach fails to consider the extent to which different system components contribute to power variations. For example, components with long idle periods may power down, and may contribute only slightly to leakage power variations. Also, for components that are mostly active, the larger contribution of dynamic power may overwhelm the variations in leakage power. Clearly, the spreadsheet-based approach, while simple and computationally efficient, cannot capture workload characteristics and component power-states. As illustrated earlier, these factors significantly influence the impact of variations on power.

4. PROPOSED METHODOLOGY

The proposed methodology for considering process variations during system-level power analysis is illustrated in Figure 3. It has three main phases. In the first phase, *leakage power modeling* is performed to obtain leakage distributions for all the power-states of the SoC components, while taking into account circuit and process characteristics. In phase 2, conventional *system simulation and power analysis* is performed to obtain a set of dynamic power and power-state traces for all components. These distributions and the power-state traces are used as inputs in phase 3, namely *Monte Carlo Analysis*. In this phase, the parameter space is sampled, leakage power is computed for all power-states and stored in a lookup-table (LUT).

This LUT, along with the traces obtained in the first phase, are used by an efficient *trace analysis* step to determine the average power consumption (considering both dynamic and leakage power) for the current sample. This step is repeated for all sample points. The larger the size of the sample space, the lower is the sampling

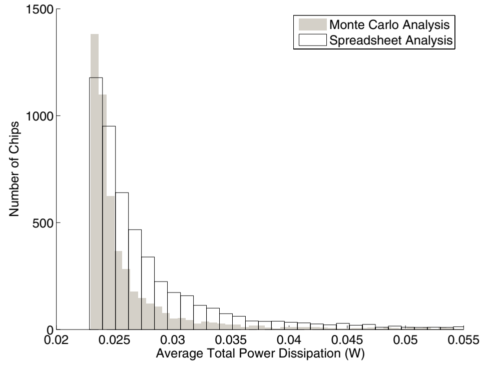


Figure 2: Comparing distributions obtained through the two analysis methods

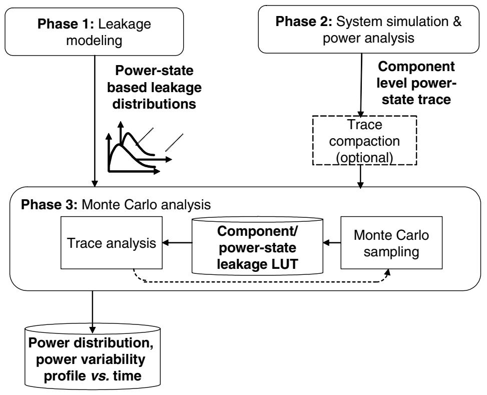

Phase 1: Leakage modeling
Phase 2: System simulation & power analysis
Power-state based leakage distributions
Component level power-state trace
Trace compaction (optional)
Phase 3: Monte Carlo analysis
Trace analysis
Component/ power-state leakage LUT
Monte Carlo sampling
Power distribution, power variability profile vs. time

Figure 3: Variation-aware system-level power analysis

error associated with the resulting power distribution. Note that, unlike the direct Monte Carlo approach described earlier, time consuming simulations are not part of the sampling loop in the proposed methodology. Therefore, very large numbers of samples can be analyzed efficiently with the proposed approach, leading to low sampling error. The power traces can be optionally processed by a *trace compaction* step prior to Monte Carlo simulations for even faster analysis. The output of the methodology is a distribution of system-level power, and optionally, a power variability profile versus time. We next describe each phase in detail.

4.1 Leakage Power Modeling

In this phase, process, circuit, and system-level characteristics are analyzed to develop variation-aware leakage power models for each SoC component. As illustrated in Section 2, for accurate incorporation of variations into system-level power analysis, it is important to consider the dependence of leakage on component power-states. A procedure to accomplish this is shown in Figure 4.

In **Step 1**, for each component, a set of power-states (*e.g.*, active, idle, sleep, deep-sleep, *etc.*) are identified. The rationale for identifying power-states is that the leakage power can be distinctly different for different power-states and depends on the number of transistors that are powered on in a particular power-state.

In **Step 2**, for each power-state of each component, circuit-level parameters are extracted for subsequent use by leakage power models. For each power-state, the total device width associated with N-type and P-type devices that are powered on is estimated. Typically at this stage of design, detailed physical implementations are not available. Therefore, high-level estimation techniques based on gate-counts (similar to the ones used in [14]) are used to estimate the total "active" device width associated with each power-state.

In **Step 3**, statistical leakage power models that consider process variations are calibrated using low-level simulation data. Most empirical leakage power models are of the form $I_{leakage} = \alpha e^{f(L,T_{ox})}$ [5, 6], capturing the exponential dependence of leakage currents on process parameters such as channel length (L) and gate-oxide (T_{ox}). Here, α depends on the gate characteristics and is directly proportional to the device width and f is a polynomial function of the process parameters and hence, is a random variable under process variations. Assuming process parameters are normally distributed, leakage current follows a log-normal distribution. The leakage of a circuit is given by the sum of the correlated log-normal

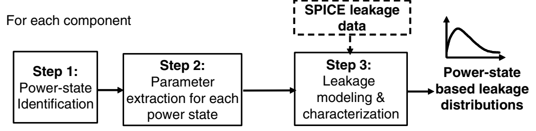

For each component
Step 1: Power-state Identification
Step 2: Parameter extraction for each power state
SPICE leakage data
Step 3: Leakage modeling & characterization
Power-state based leakage distributions

Figure 4: Power-state based leakage power modeling

leakages of its constituent gates, which is approximated with another log-normal distribution [5, 6].

The leakage current of the circuit (I_{ckt}) is sum of the leakages in the N-type transistors ($I_{n_{ckt}}$) and the P-type transistors ($I_{p_{ckt}}$). The corresponding distribution parameters are given by $\mu(I_{ckt}) = \mu(I_{n_{ckt}}) + \mu(I_{p_{ckt}})$, and $\sigma^2(I_{ckt}) = \sigma^2(I_{n_{ckt}}) + \sigma^2(I_{p_{ckt}}) + 2 * COV(I_{n_{ckt}}, I_{p_{ckt}})$. Modeling the circuit as a sea of identical gates and ignoring spatial variations within the circuit, we can write $I_{n_{ckt}} = \sum I_{n_i} = K_n * I_n$, and $I_{p_{ckt}} = \sum I_{p_i} = K_p * I_p$, where, K_n and K_p depend on circuit level characteristics identified in **Step 2**, and I_n and I_p represent the leakage current of unit width N-type and P-type devices, respectively. The distribution parameters are computed using $\mu(I_{n_{ckt}}) = K_n * \mu(I_n)$, $\sigma(I_{n_{ckt}}) = K_n * \sigma(I_n)$, $\mu(I_{p_{ckt}}) = K_p * \mu(I_p)$, and $\sigma(I_{p_{ckt}}) = K_p * \sigma(I_p)$. The parameters $\mu(I_n)$, $\mu(I_p)$, $\sigma(I_n)$ $\sigma(I_p)$ and $COV(I_n, I_p)$ are determined through SPICE simulations of accurate MOSFET models [15], where the transistor parameters are varied through Monte Carlo sampling. Similar data could also be obtained through detailed measurements of transistor off current for a set of test chips. The result of this step is a set of random variables with specified distribution parameters that model leakage power for each sub-circuit.

4.2 System Simulation and Power Analysis

In this phase (Phase 2 of Figure 3), simulation-based system-level power estimation is performed for the target SoC architecture. The inputs consist of a system-level test bench that models typical operating conditions, an architectural model of the target system, and a set of power models that track the dynamic power of each SoC component at the cycle level. The output is a set of cycle-level traces of dynamic power consumption over time for each component. In addition, a trace of the power-states for each component over time is also generated.

Trace Compaction: In this optional step, the generated traces are compacted, enabling more efficient analysis at the expense of temporal detail. The original cycle-level trace contains fields for the cycle number, dynamic power, and the power-state. High cycle-level profiling accuracy can be achieved with this trace. Trace compaction *preserving temporal information* involves collapsing consecutive cycles, in which the power-state of the component remains the same, into a single trace entry that contains the total number of collapsed cycles, average dynamic power for those cycles, and the power-state. Since temporal ordering of power-states and dynamic power is preserved, albeit at a coarser granularity, power and variability profiles *vs.* time can still be generated. If trace compaction is performed *without preserving temporal information*, more significant compaction can be achieved. The trace is reduced to a distribution of power-states (the number of clock cycles spent by each component in each state), and the corresponding dynamic power estimates that are averaged over all occurrences of each state. Using this compact representation, the analysis is extremely fast but is incapable of estimating profiles of power, or power variability *vs.* time. These schemes provide the flexibility of trading analysis efficiency for temporal resolution, based on user requirements.

4.3 Monte Carlo Analysis

In phase 3 of the methodology (Figure 3), the following two steps are executed iteratively for a fixed number of samples, to generate system-level power distributions and power variability profiles.

Monte Carlo Leakage Sampling: In this step, Monte Carlo sampling is performed on the distributions generated in phase 1 to obtain leakage power estimates for each SoC component in each of its power-states. If we are modeling only inter-chip variations, a single sample is used to determine leakage power estimates for all the components (this can be easily extended to model intra-chip variations as well). Leakage estimates are stored in a LUT that is indexed by component name and power-state.

Trace Analysis: The trace analysis procedure makes a single pass over the trace. For each trace entry, it obtains the component's leakage power from the LUT based on the power-state specified in the trace entry. The result is added to the dynamic power consumption (available in the trace). When cycle-level traces are used, this step generates a system-level power trace at the cycle level. If trace compaction preserving temporal information is used, power profiles are generated at a granularity determined by the frequency at which components undergo power-state transitions. In both the cases, average power dissipation of the system for the entire trace is calculated. When temporal information is not preserved, the procedure simply computes the average power associated with each power-state by adding the average dynamic power in that state to the corresponding leakage sample obtained from the LUT. A weighted average is then computed, using the probabilities of occurrence of the power-states as weights, to generate a power sample. By iterating the leakage sampling and trace analysis steps, we generate a distribution of total system power consumption.

5. EXPERIMENTAL RESULTS

In this section, we describe our experimental set up, and present results that compare the proposed method with the simple extensions of existing techniques described in Section 3.

Experimental Setup: For our experiments, we modeled the SoC described in Section 2 using cycle-accurate SystemC [16] models for all the hardware components, an instruction-set simulator for the CPU, and transaction-level models for the on-chip bus. All the components were enhanced with dynamic power models [13] and leakage power models (Section 4). The models were calibrated using data collected from Monte Carlo HSPICE simulations using 90 *nm* BSIM3 models [17]. We considered inter-die channel length variations with $3\sigma/\mu$ of 30%. Trace analysis was implemented using the C-MEX utility in MATLAB [18].

Efficiency Comparison: Table 1 presents the time required to generate the power distribution for the example SoC (column 2) and the resulting speed up (column 3), using different analysis methods. It can be seen that our method is about 2 orders of magnitude faster than direct Monte Carlo simulation. Efficiency improving techniques such as trace compaction enable even higher speedups of 3-5 orders of magnitude compared to direct Monte Carlo simulation, resulting in efficiency comparable to spreadsheet-based analysis.

Accuracy Comparison: Table 1 also presents the estimates of the "power yield", *i.e.*, the fraction of manufactured chips that meet a specified power budget during normal operation (columns 4-6), as estimated by different methods. Our approach results in no accuracy loss with respect to Monte Carlo simulation (rows 2-5). Rows 6 and 7 present the power yield estimates obtained using spreadsheet-based analysis. In the pessimistic approach (row 6) all transistors are assumed to be leaking all the time. Therefore, the SoC leakage distribution ($\mu = 6.36$ *mW*, $\sigma = 12.35$ *mW*) is directly added to the dynamic power estimate. In the optimistic approach (row 7) the idle components are assumed to be in deep-sleep state, thereby consuming negligible leakage power in those periods. It can be seen that the spreadsheet-based approach provides very loose quantitative bounds on the actual power yield. The proposed methodology accurately captures the times spent by each SoC component in each of its power-states, enabling a more accurate consideration of their contributions to the total system power.

Table 1: Experimental Results

Analysis Method	Efficiency Computation time		Accuracy Predicted power yield (%)		
	Time taken	Speed up	27.5 mW	30 mW	40 mW
Monte Carlo	2936m	1	80.52	87.78	95.44
Trace-based *(no compaction)*	14.49m	203	80.52	87.78	95.44
Trace-based *(Trace compaction with temporal information)*	52s	3387			
Trace-based *(Trace compaction without temporal information)*	<1s	$>10^5$			
Spreadsheet-Pessimistic	<1s	$>10^5$	64.78	76.56	92.12
Spreadsheet-Optimistic			85.56	91.66	98.18

Power Variability Profile: Figure 5 shows the temporal variation in the system-level power and its spread due to process variations, as generated by our methodology. The three waveforms from bottom to top, represent the 25th, 50th and 75th percentiles of the power distribution respectively. This information can be used to identify intervals during which the system's power variations are high, and the corresponding power-state combinations of SoC components. Such information could be used to modify the system architecture, or to design appropriate power management schemes.

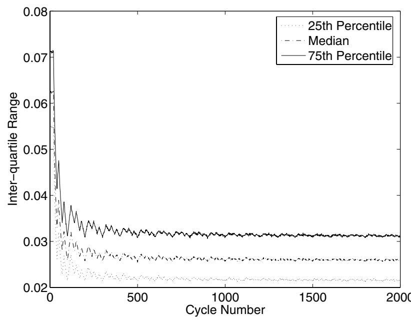


Figure 5: Power variability profile versus time

6. CONCLUSION

In this work, we demonstrated that efficient trace analysis, coupled with power-state based leakage modeling and Monte Carlo sampling provides an effective means of analyzing the impact of variations on system-level power consumption. Potential extensions of this work include the incorporation of dynamically varying parameters such as temperature and voltage, and further studies on application of such tools to variation-tolerant system design.

7. REFERENCES

[1] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proc. DAC*, pp. 338–342, 2003.

[2] L. W. Liebmann, "Layout impact of resolution enhancement techniques: Impediment or opportunity?," in *Proc. ISPD*, pp. 110–117, Apr. 2003.

[3] J. W. Tschanz et. al., "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," *IEEE JSSC*, vol. 37, pp. 1396–1402, Nov. 2002.

[4] J. A. G. Jess, K. Kalafala, S. R. Naidu, R. H. J. M. Otten, and C. Visweswariah, "Statistical timing for parametric yield prediction of digital integrated circuits," in *Proc. DAC*, pp. 932–937, 2003.

[5] R. Rao, A. Srivastava, D. Blaauw, and D. Sylvester, "Statistical estimation of leakage current considering inter- and intra-die process variation," in *Proc. ISLPED*, pp. 88–89, 2003.

[6] H. Chang and S. S. Sapatnekar, "Full-chip analysis of leakage power under process variations, including spatial correlations," in *Proc. DAC*, pp. 523–528, 2005.

[7] D. Marculescu and E. Talpes, "Energy awareness and uncertainty in microarchitecture-level design," *IEEE Micro*, vol. 25, no. 5, pp. 64–76, 2005.

[8] N. Azizi, M. M. Khellah, V. De, and F. N. Najm, "Variations-aware low-power design with voltage scaling," in *Proc. DAC*, pp. 529–534, 2005.

[9] H. Wang, M. Miranda, A. Papanikolaou, F. Catthoor, and W. Dehaene, "Variable tapered pareto buffer design and implementation allowing run-time configuration for low-power embedded SRAMs," *IEEE Trans. VLSI Systems*, vol. 13, pp. 1127–1135, Oct. 2005.

[10] `http://www.arm.com/products/CPUs/ARM946ES.html`.

[11] "AMBA 2.0 Specification." `http://www.arm.com/armtech/AMBA`.

[12] "Cell Based IC CB-90 L/M/H Type Features/Basic Specifications, NEC Electronics." `http://www.necel.com/cbic/en/cb90/cb90.html`.

[13] N.Bansal, K.Lahiri, A.Raghunathan, and S.T.Chakradhar, "Power Monitors: A framework for system-level power estimation using heterogeneous power models," in *Proc. Int. Conf. VLSI Design*, pp. 579–585, 2005.

[14] "BACPAC - Berkeley Advanced Chip Performance Calculator." `http://www.eecs.umich.edu/~dennis/bacpac`.

[15] "Industry standard deep submicron SPICE MOS device model for circuit designs." `http://www-device.eecs.berkeley.edu/~bsim3`.

[16] "The Open SystemC initiative." `http://www.systemc.org`.

[17] "BPTM - Berkeley Predictive Technology Models." `http://www-device.eecs.berkeley.edu/~ptm`.

[18] "MATLAB - High-level technical computing environment." `http://www.mathworks.com/products/matlab`.