ISSN 0002-306X. Изв. НАН РА и ГИУА. Сер. ТН. 2011. Т. LXIV, № 3.

UDC 621.382.13 RADIOELECTRONICS

S.G. ABOVYAN, G.A. PETROSYAN, A.M. POGHOSYAN, N.V. MELIKYAN

STATISTICAL STATIC TIMING ANALYSIS METHODOLOGY FOR COMPONENTS OF MICROPROCESSORS

With process technologies scaling down the process variation is becoming more important. Therefore delay variations that impact the circuit performance are also increased and affect the design timing yield. In today's microprocessors the lowest level circuit blocks contain hundred thousands of transistors, and perform statistical static timing analysis (SSTA) for them is non-trivial due to the difficulty in generating appropriate timing models. That is why statistical Monte-Carlo (MC) simulations should be run for the timing analysis of those blocks which are time consuming and almost impossible to run them on all paths in a macro or even on a few top critical paths. The proposed fast and precise methodology can be used to run SSTA on the lowest level circuits of microprocessors. It provides about 90% accuracy and 5-10 times runtime saving compared to MC results.

Keywords: statistical static timing analysis, microprocessors, lowest level circuits, Monte-Carlo simulations.

Introduction. Technology scaling has brought the rapid increase in process variability [1]. Its effects on device performance have compelled the industry to transition to statistical techniques for timing sign-off. Traditional corner case analysis (CCA) [1,2] constrains the design and often sets stringent, unrealistic timing specifications. Moreover, for technology nodes smaller than 65 *nm*, these overestimated timing bounds compensate the performance improvement due to device scaling. SSTA [2] is used in practice to analyze the impact of process variations on timing. It handles the random parts of the process variations as probability distributions to calculate the delay statistically. SSTA has gained widespread acceptance for standard cell based designs, as it removes a significant portion of pessimism introduced by conventional approaches like CCA while accounting for global (inter-chip) and local (intra-chip) process variations [3]. The application of both path based and block based SSTA have been shown to be advantageous [2] for cell based ASICs for which reusable timing models could be easily characterized. The method of SSTA for microprocessors is proposed in [4] which is applicable only for standard cell based blocks. But, for example, cache blocks in microprocessors are not made of standard cells. More than 50% of a multi-core processor and more than 30% of each core are occupied by cache arrays and custom, transistor level blocks both of which are not standard-cell based. For custom macros, there are significantly more transistor level options to improve performance with less overhead than in case of gate level circuits. Moreover, such macros occur in portions of the processor which are extremely timing critical where variations could adversely affect the final performance.

The methodology proposed in [5] for generating statistical models for the large IP macros can be used in SSTA flows allowing fast analysis. While this method is shown to be accurate, it works only for macros with gates as basic units and cannot be easily adapted for transistor level macros. A method of variation aware transistor level timing analysis for macros is described in [6]. Statistical models are built for macros at a chip level of hierarchy. These

approaches introduce some inaccuracy in predicting chip level performance degradation due to variations. To overcome these problems and to perform accurate variation analysis of transistor level macros, rigorous, but time consuming MC SPICE simulations of selected paths are currently used. The simulation run time is of the order of hours/path. It is impractical to perform such MC simulations on all paths in the macros and is therefore required to have a prior knowledge of the top paths that could potentially become critical. Hence it becomes necessary to have a fast statistical timing analysis flow for transistor level macros that can compute the delay distributions due to process variations of all paths in the macros with accuracy close to MC simulations.

The proposed methodology finds a solution to this problem. It first groups the macro transistors into logic gates called xcells by applying special grouping technique which does not approximate any transistor or wire information. It is vital in preventing any accuracy loss. For all extracted xcells timing library considering both inter-chip and intra-chip process variations using a SPICE circuit simulator is built. The library is later used by an industrial-standard timing engine to perform block based SSTA of the macro.

Global and local process variations. Threshold-voltage (V_{th}), effective channel length (L_{eff}), oxide thickness (T_{ox}), mobility (μ), and dopant concentration (C) are the main variation parameters that significantly affect performance. Their variations result in designs with a wide spread of critical path delay distributions that may degrade the timing yield, i.e. decrease the fraction of manufactured chips that meet the timing constraints. For analysis purposes, parameter variations are usually classified into two categories: the inter-chip or global and the intra-chip or local variations. In case of globally varying parameters, their values are the same for all devices on the chip.

Variation parameters may depend on each other. For instance, an increase in T_{ox} also increases V_{th} . Principal component analysis is used to convert the dependent variation parameters into independent principal components (PCs). In general, the delay of a path *D* due to variation is given by [7]:

$$D = D_0 + \sum_{i=1}^{n} \sigma_{p_i} \cdot Z(Y_i) + \sum_{i=1}^{n} \sum_{k=1}^{j} \sigma_{m_{ik}} \cdot Z(L_{ik}) , \quad (1)$$

where D is the path delay; D_0 is the nominal delay (without variation); σ_{p_i} is the standard deviation of the delay distribution due to the global random variable $Z(Y_i)$; i varies from 1 to n - number of principal components; $\sigma_{m_{ik}}$ is the standard deviation of the delay distribution due to the local random variable $Z(L_{ik})$; k varies from 1 to j - number of transistors.

In equation (1) the local delay component is dependent on the number of transistors. The fact that for global variations all transistors within a macro are completely correlated and for local variations they are completely uncorrelated (statistically independent) helps re-write equation (1) as follows [7]:

$$D = D_0 + \sum_{i=1}^{n} \sigma_{p_i} \cdot Z(Y_i) + \sum_{i=1}^{n} \sigma_{m_i} \cdot Z(L_i) . \quad (2)$$

In equation (2) the number of local random variables Z(L) is reduced from nj to just n showing that Z(L) does not depend on the number of transistors in the macro. This is a useful result because in a macro, the number of transistors j could be in millions.

Proposed approach. The proposed SSTA flow developed for transistor macros is shown in Figure 1. It consists of three major steps.

- Transistor level macro is converted to gate level blocks called xcells using special grouping procedure.
- Variation aware library is characterized for these xcells using the variation aware SPICE models.
- Block-Based SSTA analysis are executed [4].

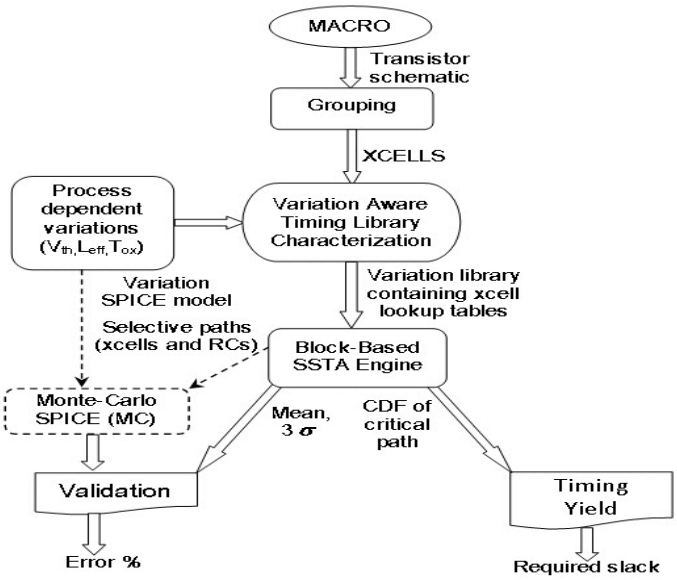

Fig. 1. Proposed SSTA flow

The method used by block-based SSTA engine in Figure 1 is described in [4]. It is based on simultaneous application of the usual static as well as statistical static timing analysis. At the first stage usual static timing analysis (STA) is applied and at the second stage - SSTA. The offered method of the analysis allows to reach acceptable analysis results from the practical point of view of accuracy at rather small expenses of machine runtime. SSTA engine determines delay distributions for all paths in the macro using the variation libraries considering equation (2). The validation step compares the SSTA results with MC results. The timing yield step estimates the required arrival time based on the most critical path due to variation.

Convert a macro to the xcells. The conversion of the transistor level netlist into a netlist of xcells is performed by using special grouping technique which is developed to facilitate hierarchical, transistor level static timing analysis using industrial block-based timing

analyzers. It takes as input a transistor level GDSII layout of a macro and obtains a logic (verilog format) and parasitic netlists (spef format) as outputs that can be used by a static timing engine. The logic netlist consists of xcells each of which contains transistors that are source/drain connected to its output node.

The xcells are inferred by a rule-based recognition process that can recognize static CMOS, transmission gates, cross-coupled domino gates, latches, and flops. Using the inherent hierarchy in memory blocks like cache, specialized xcells are formed by grouping a number of SRAM bit cells (~5000 bit-cells per xcell) that are referred to bit-columns. The parasitic netlist contains the interconnect and device internal parasitics. The latter include the transistor parasitics that are pushed to the output node of each xcell. In order to reduce the number of inferred xcells that must be characterized, the xcells that have the same topology and whose internal parasitics are within a small range are folded to form a single xcell. An average xcell other than the bit-column typically consists of 10…15 transistors.

Variation library characterization. After converting a macro to gate level xcell netlist, a timing library is generated. It contains delay/output slew look-up tables for each pin in the xcell and for all PCs. This is accomplished using an automated characterization engine [8] that performs SPICE simulations to obtain delays for a wide range of input and output conditions (slew/load).

Each xcell in the library is characterized at $2N+1$ different values of the PCs stored as $2N+1$ look-up tables; N is the number of PCs. The characterization process is performed for 10 PCs where each xcell has 21 tables in the library. One table corresponds to the nominal case, with all 10 PCs set to their mean (nominal) values. The other 20 tables are generated for xcells characterized at the $+3\sigma$ and -3σ values for the 10 PCs.

Results. A 45 *nm* design macro for experiments is used. It contains 60 unique xcells. The total number of transistors in the macro is of the order of a few hundred thousands. The delay values shown in the figures are normalized to 500 *MHz*.

It is important to mention that simulation results depend on technology and number of used xcells. Increased number of xcells will increase simulation time almost linearly for the same technology.

Monte-Carlo Vs SSTA. The studied macro has the critical path that requires at least 3 hours to complete MC. This makes it impractical to perform MC using variation device models for all top paths in the design. SSTA allows to see these distributions [9] and hence analyze the effects of variation on all paths of the design which is the most important goal achieved. Figure 2 shows slack values of the top critical paths in the design.

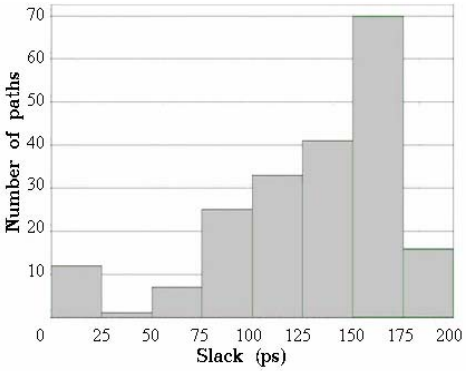

Fig. 2. Slack values of some of the top critical paths

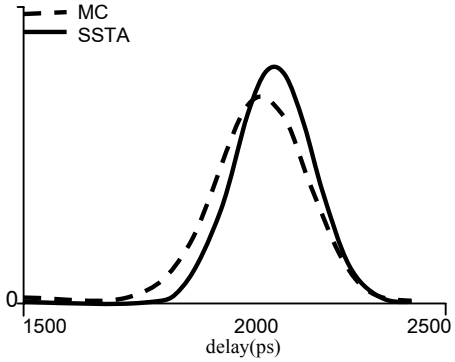

Fig. 3. Comparison of the critical path's delay distributions obtained by MC simulations and SSTA flow

In order to run MC to validate SSTA, 50 paths of different lengths in terms of xcell number are pruned out from the macro netlist. A few representative paths are listed in Table 1. The extracted layout parasitics are also included during MC simulations. Table 1 compares the mean and the standard deviation (1σ) of the endpoint delays (arrival time) between SSTA distributions and distributions obtained after 1000 runs of MC simulations. Figure 3 compares the delay distributions of the most critical path in the macro obtained by MC simulations and proposed SSTA flow.

The maximum error percentage of the total variation of SSTA reported delay is ~6%. The table also shows the runtime for MC simulations. The runtime for the entire SSTA, which computes the distributions for all paths in the macro, is almost negligible, less than 3 minutes for a macro of ~100,000 transistors.

It is very important to mention that Table 1 does not include library characterization runtime. The library characterization time varies within 2 ~ 8 hours. As it is shown in Table 2, for a 5% compromise in accuracy the library characterization time can be significantly reduced.

Table 1

Comparison of MC and SSTA path delays

Xcells /path	Monte-Carlo SPICE (MCS) Simulation			SSTA		Error
	Runtime	Mean (μ1) Delay (ps)	Total (T_1) (1σ Delay) (ps)	Mean (μ2) Delay (ps)	Total (T_2) (1σ Delay) (ps)	$\frac{\mu_1+\sigma T_1}{\mu_2+\sigma T_2}$
15	2 hrs	213	9.24	205	8.2	4.2%
4	25 min	70.5	2.65	77.2	0.4	5.7%
8	50 min	16.3	1	17	1.4	6.0%
13	1.5 hrs	291	12	299	10	1.9%
5	35 min	45.5	4.3	44.3	3.7	3.7%

Timing Yield. Without SSTA it is necessary to fix the critical paths to meet a frequency that is much greater than the target frequency needed for a particular yield.

Figure 4 shows the CDF of the most critical path the period of which is define by the frequency of the entire macro. 50% yield point corresponds to the nominal time period of 2000ps at which the SSTA is performed for this macro. For instance, if needed to achieve a 68% yield at 2000ps, SSTA results suggest a minimum required arrival time (RAT) of 2011ps to be set on the critical path based on the slack difference. This design has a large positive slack of 53ps even for a 98,5 % yield, suggesting that the design has been over-optimized. Figure 5 compares the slack values obtained for all paths by setting the minimum RAT from SSTA at 60% and 90% yield points and a conservative RAT used to fix the design before using SSTA flow. Figure 5 shows a clearly large margin that is pessimistic even to achieve a 90% yield.

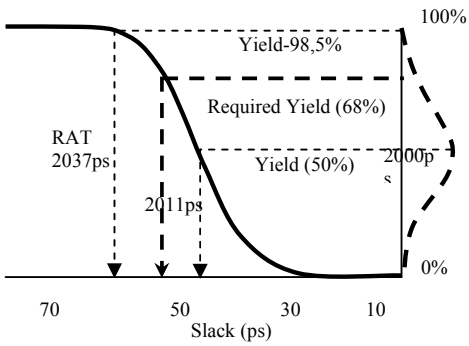

Fig. 4. Timing yield plot – CDF of the most critical path of the design obtained using SSTA with RAT = 2000ps

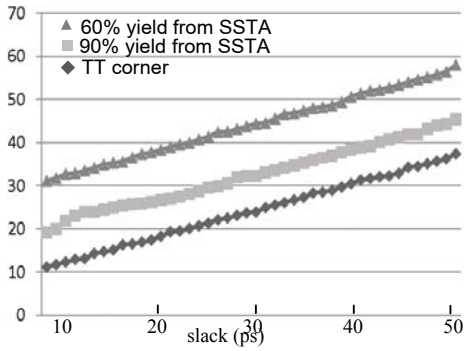

Fig. 5. Slack values of all paths of the design obtained by performing SSTA with RAT chosen from 60%, 90% yield points and conventional corner case TT

Characterization runtime. Characterization of a variation library at $2N+1$ points as described above for each xcell even though is a one-time effort, is still time consuming. However, libraries generated this way for different PC corners are more accurate, since the sensitivities are determined from look-up table delay values obtained by actual circuit simulation rather than analytical formulations. The library generation time linearly increases with the number of points at which each xcell in the library is characterized. For each point of characterization, a look-up table is generated for every xcell in the library

Table 2

Characterization runtime reduction

Number of tables/xcell	Accuracy	Runtime
Characterization 1: 21 tables	100% (normalized)	~8hrs
Characterization 2: 11 tables	97%	~4hrs
Characterization 3: 5 tables	95%	~2hrs

Characterization 1. Considered all 10 PCs.
Characterization 2. Only considered 5 global PCs and set a correlation of 1 between transistors to represent global variations. The same PCs are used setting a correlation of 0 between transistors to represent local variations.
Characterization 3. Assuming the delay variance obtained for each PC variation is symmetrical about the mean delay value, only $N+1$ tables instead of $2N+1$ for the 5 global PCs are characterized. The xcells are characterized only at the mean -3σ points of the PCs.

Conclusion. New SSTA methodology for microprocessors which execute timing analysis for transistor level and provides distributions for all paths in the macro that are close to MC results (~90% accuracy) is proposed. The methodology also helps pin-point the paths and their components that are more sensitive to a particular source of process variation (V_{th}, T_{ox}, μ, L_{eff}) which can be used for design optimization.

While this flow is developed mainly for transistor macros, it can easily be modified to be used for any cell based macro.

References

1. **Chiang C., Kawa J.** Design for Manufacturability and Yield for Nano-Scale CMOS - Springer, Dordrecht, The Netherlands, 2007.
2. **Blaauw D., Chopra K., Srivastava A. and Scheffer L.** Statistical Timing Analysis: From Basic Principles to State of the Art // IEEE transactions on computer-aided design of integrated circuits and systems.- 2009.- Vol. 27. P. 589-607.
3. Process Variation Statistical Modeling for VLSI Timing Analysis / **J. Liu, J. Zeng, A. Hong et al** // 9th International Symposium on Quality Electronic Design.- 2008.- P. 730-733.
4. **Меликян В., Абовян С., Петросян Г.** Статистический анализ временных задержек мультипроцессорных систем // Электроника и связь.- Киев, 2010. -Т.5, №58. -С. 108-112.
5. **Goel, A., Vrudhula, S., Taraporevala, F., Ghanta, P.** A Methodology for Characterization of Large Macro Cells and IP Blocks Considering Process Variations // Proc. of 9th International Symposium on Quality Electronic Design.- March, 2008.- P. 200–206.
6. A hierarchical transistor and gate level statistical timing analysis flow for microprocessor designs / **Sinha, D., Bhanji, A., Visweswariah, C. et al** // Proc. Design Automation Conference, session 4u.3s.- July, 2009.
7. **Sundareswaran, S., Abraham, A., Panda, R., and Ardelea, A.** Characterization of standard cells for intra-cell mismatch variations // IEEE Transactions on Semiconductor Manufacturing.- Feb. 2009.- Vol. 22 (1).- P. 40-49.
8. **Synopsys Liberty NCX User Guide**, 2009.
9. **Synopsys PrimeTime-VX User Guide**, 2010.

SEUA. The material is received on 25.04.2011.

Ս.Գ. ԱԲՈՎՅԱՆ, Գ.Ա. ՊԵՏՐՈՍՅԱՆ , Ա.Մ. ՊՈՂՈՍՅԱՆ, Ն.Վ. ՄԵԼԻՔՅԱՆ

ՄԻԿՐՈՊՐՈՑԵՍՈՐՆԵՐԻ ՀԱՆԳՈՒՅՑՆԵՐԻ ՎԻՃԱԿԱԳՐԱԿԱՆ ՍՏԱՏԻԿ ԺԱՄԱՆԱԿԱՅԻՆ ՎԵՐԼՈՒԾՈՒԹՅԱՆ ՄԵԹՈԴ

Տեխնոլոգիական գործընթացների մասշտաբավորման հետևանքով պարամետրերի շեղումները դառնում են առավել կարևոր: Հետևաբար սխեմայի արտադրողականության վրա ազդող հապաղման շեղումները նույնպես դառնում են նշանակալի և ազդում պիտանի ելքի տոկոսի վրա: Ժամանակակից միկրոպրոցեսորներում ստորին մակարդակի սխեմաներով հանգույցները պարունակում են հարյուր հազարավոր տրանզիստորներ, և համապատասխան ժամանակային մոդելների ստեղծման բարդության պատճառով վիճակագրական ստատիկ ժամանակային վերլուծություն (ՎՍԺՎ) կատարելը դրանց համար դառնում է գործնականում անհնարին: Այդ պատճառով այսպիսի հանգույցների ժամանակային վերլուծության համար իրականացվում է վիճակագրական Մոնտե-Կառլո մոդելավորում: Այդ մոդելավորումները ժամանակատար են, և գրեթե անհնար է դրանք կիրառել բոլոր կամ նույնիսկ մի քանի կրիտիկական ուղիների վրա: Առաջարկված է արագ և ճշգրիտ մեթոդ, որը կարող է օգտագործվել միկրոպրոցեսորներում ստորին մակարդակի սխեմաներով հանգույցների վրա ՎՍԺՎ իրականացնելու համար: Այն Մոնտե-Կառլո մոդելավորման հետ համեմատած ապահովում է մոտավորապես 90% ճշտություն՝ ծախսելով 5…10 անգամ ավելի քիչ ժամանակ:

Առանցքային բառեր. վիճակագրական ստատիկ ժամանակային վերլուծություն, միկրոպրոցեսոր, ստորին մակարդակի սխեմաներ, Մոնտե-Կառլո մոդելավորում:

С.Г. АБОВЯН, Г.А. ПЕТРОСЯН, А.М. ПОГОСЯН, Н.В. МЕЛИКЯН

МЕТОД СТАТИСТИЧЕСКОГО СТАТИЧЕСКОГО ВРЕМЕННОГО АНАЛИЗА МИКРОПРОЦЕССОРНЫХ УЗЛОВ

С уменьшением размеров технологий вариации параметров становятся более значительными. Следовательно, вариации задержек, воздействующие на работоспособность схемы, также возрастают и влияют на производственный выход. В современных микропроцессорах блоки схем нижнего уровня имеют сотни тысяч транзисторов, и выполнение статистического статического временного анализа (ССВА) становится практически невозможным из-за сложности создания соответствующих временных моделей. Поэтому для временного анализа этих блоков должны выполняться статистические симуляции Монте-Карло. Отмеченные симуляции проводятся довольно долго, и их почти невозможно выполнять для всех или даже для нескольких критических путей. В данной работе предлагается быстрый и точный метод, который может быть использован для ССВА схем нижнего уровня в микропроцессорах. По сравнению с анализом Монте-Карло, предлагаемый метод обеспечивает приблизительно 90% точности и сокращение машинного времени в 5…10 раз.

Ключевые слова: статистический статический временной анализ, микропроцессоры, схемы нижнего уровня, симуляции Монте-Карло.