A Real Delay Switching Activity Simulator based on Petri net Modeling

Ashok K. Murugavel and N. Ranganathan
Center for Microelectronics Research
Department of Computer Science and Engineering
University of South Florida, Tampa

murugave,ranganat@csee.usf.edu

Abstract

Switching activity estimation is an important step in power estimation of digital VLSI circuits. While simulation yields accurate results, it is time consuming. In this paper, we propose a new technique based on Petri nets for real-delay switching activity estimation that yields the same accuracy as simulation, but is significantly faster in computation. We introduce a new type of Petri net called Hierarchical Colored Hardware Petri Net (HCHP-Net). The gate-level circuit is first transformed into a directed acyclic graph called, GSDAG, in which both the gates as well as the signals correspond to the nodes in the graph. The GSDAG is then mapped onto a corresponding HCHP-Net which is then simulated using a Petri net simulator. Experimental results for ISCAS '85 circuits are presented. The method replicates exactly the switching activity results for real-delay models produced by HSPICE and PowerMill. However, the per-pattern simulation time is about 51 times faster than the Synopsys PowerMill and 8900 times faster than the Avanti HSPICE.

1 Introduction

Power consumption in CMOS VLSI circuits has added a new dimension to the design space. In addition to area and time, power has become an important design parameter. The advent of portable, wireless computing has increased the need for low power high performance compute intensive VLSI circuits. Power estimation at high levels of abstraction is fast but at the cost of accuracy, at low levels of abstraction, the speed is slower but accurate. The power estimation techniques in general reflect a trade-of between accuracy and speed. Power consumption in CMOS circuits can be classified into two components, (i) static and (ii) dynamic component. The dynamic component can be further classified into two components: (a) short-circuit and (b) switching component. The short circuit and the static components of power can be made negligible by good selection of devices and using a well designed circuit. The power equation for the switching component can be given as $P(x) = \frac{1}{2}V_{dd}^2 f C \alpha(x)$, where V_{dd} is the supply voltage, f is the operating frequency, C is the output load capacitance and $\alpha(x)$ is the switching activity at the output of node x. The switching activity at each node can be estimated knowing the structural details of the circuit obtained either from a gate-level net-list or from a structural VHDL code. In this paper, we propose a new simulative method for switching activity estimation of circuits modeled and simulated as Petri nets. While the proposed method yields same results as known commercial simulators such as Avanti HSPICE and Synopsys PowerMill, it is much faster in terms of execution time.

2 Related work

Numerous works have appeared in the literature on average power estimation. The methods for average power estimation can be broadly classified as *(i) simulative* and *(ii) non-simulative*. In simulative approaches, the circuit is simulated for different input sequences and the average power is calculated as an average of the simulated values. Exhaustive simulation is accurate and takes care of spatial and temporal correlations within the circuit, however it can be time consuming. The non-simulative approaches can be classified as *(i) probabilistic* and *(ii) statistical*. In probabilistic techniques, the input probabilities are propagated through the circuit and the switching probability at each node is computed. Probabilistic techniques are weakly pattern dependent. They are fast and tractable but typically involve assumptions about joint distributions. In statistical techniques, a circuit is simulated for a small set of vectors and the output is monitored. A stopping criteria is used to determine whether the simulations are to be stopped. Examples of power estimation works that assume a zero delay

model can be found in [2, 10, 16, 17, 21]. A few probabilistic and statistical methods for real delay power estimation have been presented in [5, 12, 19, 22, 18]. When characterizing modules for RT-level power estimation, the accuracy of the power estimate is important, making simulation approaches still beneficial. Next, we briefly visit related works in real-delay simulation for power estimation.

In [1], a new methodology based on the current limited model is proposed. The maximum error in this methodology was about 10%. In [20], a switch level simulation based methodology has been presented. In [3], a novel event driven delay simulation method using a time parallel simulator is described. A new glitch modeling technique for logic simulation is presented in [13, 14]. The most important features of this model are the enhanced scheduling of glitch events and the prediction of peak glitch voltages. The source of error in accuracy is the error in activity estimation, which increases dramatically with circuit depth. In [9], a fast and accurate estimate of the average power consumed by the circuit based on a toggle count mechanism has been presented. The toggle information obtained is not exact for circuits with high depth. This leads to discrepancies in the power estimate. The methodology has been compared with HSPICE and PowerMill.

In this paper, we propose a new technique based on Petri nets for real-delay switching activity estimation. Petri nets have developed over the last two decades for representing and analyzing concurrent systems. A Petri net is a marked directed graph whose nodes represent either a place or a transition. This structure essentially models the gate-level circuit description in a compact manner taking into consideration, the various correlations that occur in a circuit. The first step is to construct a Directed Acyclic Graph (DAG) based on the logical structure of the circuit. Each gate and fan-out net in the circuit is modeled as a node in the DAG. We call this DAG as Gate Signal Directed Acyclic Graph (GSDAG). The GSDAG is converted into a Petri net for simulation.

3 Petri Nets Basics

Petri nets consist of set of places (P) and a set of transitions (T). "*The states of the model are represented as places, the passive components*" and "*The action oriented to the states of the model are represented as transitions, the active components*". The places and transitions are connected by *arcs* (A). The arcs are directed and can only connect a transition with a place or vice-versa. A transition is said to be *enabled* if there are enough tokens in each of the input places as specified by the arcs connecting the input places to the transition. An enabled transition can *fire* if the other conditions associated with the transition are satisfied. Using the concept of Colored Petri nets (CP-nets) from [7], Hierarchical Colored Petri nets (HCP-nets) from [6] and Hardware Petri nets (HP-nets) from [15], we define a new kind of Petri net called Hierarchical Colored Hardware Petri nets (HCHP-nets), so we give a brief overview of CP-nets, HCP-nets and HP-nets first. HCP-nets are an extension of CP-nets and so the definitions for CP-nets also hold for HCP-nets. The three basic elements of CP-nets are places, transition and arcs. The current state of a Petri net is the distribution of tokens to the places in the Petri net. An arc directed from a place to a transition and from a place to a transition is called an input-arc and output-arc of a transition respectively. The initial state of the CP-net is called the *initial-marking*. CP-nets in addition have *type* and *color* (C) information attached to places and tokens in the net. The arcs have expressions attached to them called *arc-expressions* (E). A transition in a HCP-net is enabled if it is possible to bind the variables in such a way that the arc-expressions of all the input arcs evaluate to tokens present at the corresponding input places. The transition can in addition have Boolean expressions called *Guard-expressions* (G), that need to evaluate to "true" in-order that the transition fires. It is difficult to model large systems as a single large CP-net. To alleviate this problem the concept of Hierarchical Colored Petri nets [6] is being used. The concept of Hardware Petri nets (HP-nets) [15] has been used to model asynchronous circuits. Now to define a HCHP-net we need certain mathematical terms such as multi-sets, Time sets and relation.

Definition 1: [8] A *multi-set* s, over a set S, is a function from S to N, where N is the set of Natural numbers. If $a \in S$ then $s(a)$ is the number of appearances of element a in s. A_{MS} is the set of all multi-sets over S. Formally defined as,

$$\sum_{a \in S} s(a)a \tag{1}$$

Definition 2: A Time set (TS) is defined as the set of all non-negative real numbers, $TS = x \in R$, where x represents the time instant and R is the set of all real numbers.

Definition 3: For each transition $t \in T$, Var(t) is the set of variables associated with transition t given as,

$$\forall t \in T : Var(t) = \{v | v \in Var(G(t)) \vee \exists a \in A(t) : v \in Var(E(a))\} \tag{2}$$

where, v is the variables in the Petri net, $Var(G(t))$ is the set of variables associated with the Guard-expression (G), a is the arcs in the Petri net, $A(t)$ is the set of arcs associated with transition t and $E(a)$ is the arc-expressions associated with arc a.

4 Hierarchical Colored Hardware Petri Nets

In this work, we have modified the HP-nets and HCP-nets to define a new kind of Petri nets as Hierarchical Colored Hardware Petri nets (HCHP-nets). In order to obtain the switching activity of a circuit under a real-delay model, we need to add the factor of time in to the Petri net. The important feature that need to be distinguishing here is real-time and simulated time. *Real Time*: Real world physical time in which the simulator executes. *Simulated Time*: The time that is built into the model to incorporate timing features. The features of HCHP-nets are: (i) each of the places can store only one token at any point in time. (ii) when a transition fires, the operation that is associated with the gate that is being fired is performed. (iii) besides the weight on the arcs, functions are defined on the arcs to create additional firing conditions related to the markings on the places they are connected with. Additional firing conditions are attached to each transition to specify delay equivalent of the gates or the interconnect. (iv) each token has a time stamp and is called a timed token. The timed token is available only after the clock time of the simulator is greater than or equal to the the time stamp on the token.

The HCHP-nets in addition to the components defined in the previous section has a set of *substitution transitions*, that relates a simple transition to a more complex CP-net. A non-hierarchical CP-nets in a HCHP-net is called *page*. Each non-hierarchical CP-net (page) has a set of input and output *ports* that are places which specify the input and output types of the page. The places connected to the substitution transition in a hierarchical CP-net, must be related to the port places in the non-hierarchical CP-net. HCHP-nets can be defined mathematically as,

Definition 4: A HCHP-net is defined as a tuple,

$$HCHPN = (PG, S, PO, PA, PR, TI, TO, TC) \quad (3)$$

where,

(i) PG is a finite set of pages such that each of the pages is a non-hierarchical CP-net, and none of the pages have any net element in common,

(ii) $S \subseteq T$ is a set of substitution transitions,

(iii) $PO \subseteq P$ is a set of port nodes. A port node is a place in the page with either a set of input-arcs or a set of output-arcs,

(iv) PA is a port assignment function. It defines the relation between the socket nodes (the places attached to the substitution transition) with the port nodes of the page,

(v) $PR \in PG_{MS}$ is a multi-set of prime pages. It determines the number of instances each of the individual pages has,

(vi) TI is the Time set as defined in **Defn. 2**,

(vii) TO is the set of all possible colored tokens, i.e., all triples (p, d, t), where $p \in P$, d is the value of the token and $t \in TI$,

(viii) TC is the code segment function. It is defined from T into the set of functions.

The 8-tuple (PG, S, PO, PA, PR, TI, TO, TC) specifies the structure of the HCHP-net. Now, we discuss the behavioral characteristics of the HCHP-net.

Definition 5: A *binding* of a transition t is a function defined on Var(t); as,
$(i) \forall v \in Var(t) : s(v) \in Type(v)$,
$(ii) G(t) < s >$
B(t) denotes the set of all bindings for t as defined by the function s.

Definition 6: A *token element* is a tuple (p,c,t) where $p \in P$, $c \in C(p)$ and $t \in TI$, while a binding element is a pair (t,b), where $t \in T$ and $b \in B(t)$. The set of all token elements is denoted by TE while the set of all binding elements is denoted by BE.

Definition 7: A *state* is defined as a multi-set of colored token with a time-stamp. S is the set of all possible states. A *marking* is a multi-set over TE without the time-stamp. The initial marking M_0 is the marking which is obtained by evaluating the initialization expressions:

$$\forall (p, c) \in TE : M_0(p, c) = (I(p))(c) \quad (4)$$

The sets of all markings is denoted by M.

Definition 8: *Event* (t, a, b) specifies the possibility of the firing of a transition t by removing tokens from the input places a and adding tokens to the output places b. The set of all events is the Event set ES.

Definition 9: A step S is *enabled* in a marking M iff

$$\forall p \in P : \sum_{(t,b) \in Y} E(p, t) < b > \leq M(p) \quad (5)$$

Let the step S be enabled in the marking M. When (t,b) $\in$ S, we say that t is enabled in M for binding b. When $(t_1, b_1,), (t_2, b_2) \in Y$ and $(t_1, b_1) \neq (t_2, b_2)$ we then say that (t_1, b_1) and (t_2, b_2) are concurrently enabled.

After the transition is enabled, it is not necessary that the transition must fire, this is because of the time factor involved, which leads to the definition of enabling time.

Definition 10: The *enabling time* E of an event (t, a, b) is the maximum of the time-stamps of the tokens consumed by transition t from the input place set a.

$$E(t, a, b) = max_{(p,d,t) \in TO} x \quad (6)$$

Definition 11: The *enabling time* of a transition is the maximum time stamp of the tokens to be consumed.

Definition 12: If a transition t is enabled and the global clock is past the enabling time, it can *fire*. If more than one transition can fire at the same time, the priority of the transition firing is given to the transition that was enabled

first. The firing of transition t changes the marking **M** to marking **M'** such that,

$$M'(p) = \begin{cases} M(p) \pm aw & if(p,t) \in A, \\ M(p) & otherwise \end{cases}$$

where $M(p)$ and $M`(p)$ is the marking of place p before and after the firing of transition t. The value aw is the weight of the arc from place p to transition t.

5 Switching Activity Modeling for real-delay using Petri Nets

To model switching activity estimation using Petri nets, we first represent the the gate-level combinational circuit as a GSDAG. We illustrate this with an example using c17 an ISCAS circuit with 6 gates as shown Figure 1.

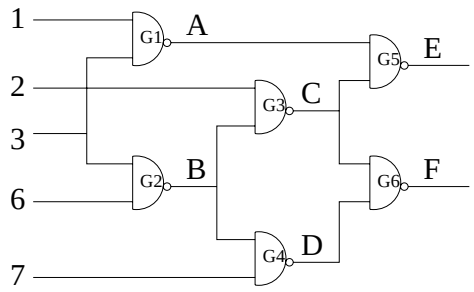

Figure 1. c17 ISCAS Combinational Circuit

The circuit shown in Figure 1, has 6 gates marked *G1, G2, G3, G4, G5, G6* and we are interested in capturing the switching activity at the output of each gate marked *A, B, C, D, E, F*. In order to model the switching activity at each of the gates outputs we need to consider the input to each of the gates. First we represent the circuit as a DAG where each gate in the circuit is represented as a node and each signal in the circuit as an arc. To calculate the switching activity in the circuit we need to calculate the number of times the output of the gate switches from either 0 to 1 or 1 to 0. The DAG thus obtained is represented as a GSDAG structure, where in addition to the nodes for each gate, each signal is also a node. The GSDAG corresponding to the circuit shown in Figure 1 is given in Figure 2. To represent multiple fan-outs, we have a node for each output from the gate. Each of the gates in the GSDAG is replaced by a transition and the signals are replaced by a place. The new structure thus formed is a Petri net.

The Petri net thus obtained from the GSDAG is transformed into a a HCHP-net. The places in a HCHP-net have a type function specifying the type of tokens the place is associated with. Each transition is either associated with a SML [11] code or is replaced as a substitution transition

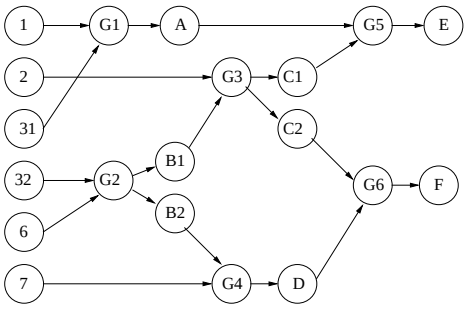

Figure 2. GSDAG structure of the c17 circuit

to perform the corresponding gate function. Figure 3 corresponds to the the HCHP-net equivalent of the ISCAS c17 combinational circuit. If the Petri net obtained from the GSDAG is large, they are segmented.

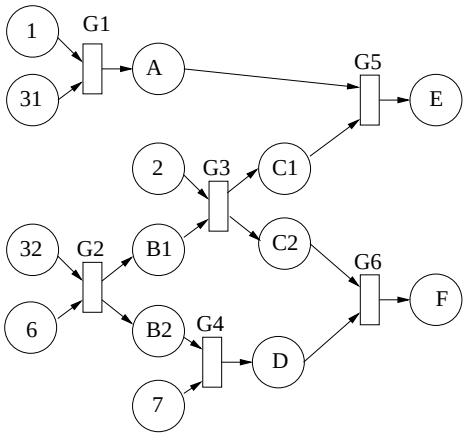

Figure 3. Petri net structure of the c17 circuit

Simulation is a technique for analyzing a system using controlled experiments. It is a very powerful technique and doesn't have any additional restraints. Since we are using Petri nets to actually study the properties in a circuit simulation analysis of the HCHP-nets, it is the fastest and best method to estimate the switching activity in the circuit. To calculate the toggle information, we need to calculate the various instances at which the gate switches from either 0 to 1 or from 1 to 0. A transition fires in a HCHP-net only when an event occurs i.e., the transition if enabled for firing. The coded segments added to the transition are used to determine the occurrence of a switch. The information obtained from the transition firing specifies the number of times the transition occurs at the output of a gate. The information from all the gates is calculated to determine the switching activity of the circuit. Thus we get a direct relationship between the transition firing in the Petri net and the switching activity in the circuit.

Figure 4 gives the block diagram of the switching activ-

ity estimation tool using HCHP-nets. The input to the tool is the gate-level net-list of the circuit whose switching activity needs to be estimated, the accurate delay for each of the gates being used in the circuit and the input vectors to be simulated. The first step in the tool is the conversion of the net-list into a GSDAG. The GSDAG thus obtained is now passed on to the second step, where the GSDAG is converted into a HCHP-net with the necessary code segments, substitution transition, color functions, hierarchy and delay element. The HCHP-net thus obtained is given to the Design/CPN tool [11], to perform the simulation. The toggle information is thus generated, which is the switching activity of the circuit. In the next section experimental results for various benchmark circuits have been provided and a comparison of the time taken for the simulation of different vector sizes is given.

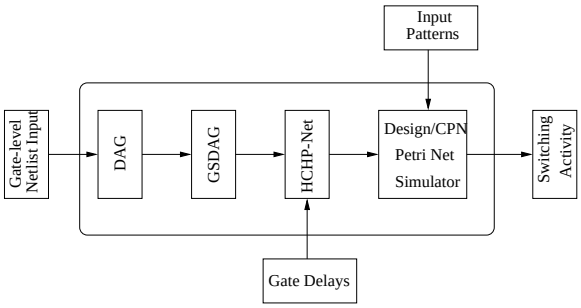

Figure 4. Block diagram of switching activity estimation tool using HCHP-nets

6 Experimental Results

The complete ISCAS '85 combinational benchmark circuits have been mapped to the proposed HCHP-nets. The average switching activity in each of the circuits has been estimated as specified in the previous sections. We have used Design/CPN [4, 11], a Colored Petri net simulation and analysis tool for performing the simulation of the HCHP-nets. Also, the proposed methodology is compared with other circuit level simulators like HSPICE and PowerMill. The ISCAS '85 suite of benchmark circuits are combinational circuits consisting of the basic gates, like AND, OR, NAND, NOR, NOT, EXOR, BUFF. The ISCAS circuits include gates with a large number of inputs, these gates have been modified to include gates with only two, three and four input gates, and preserving the logic structure in the original gate. The switching activity at the output of each of the gates is thus accurately calculated. *It is important to note that the switching activity values generated by our method produced the exact same results as those from HSPICE and PowerMill.* This indicates the accuracy of the proposed approach. SPICE files were generated for the modified ISCAS '85 benchmark circuits to be given as input to HSPICE and PowerMill.

Table 1. Simulator Performance

Circuit	HSPICE (in ms)	PowerMill (in ms)	HCHP-nets (in ms)	Speed-up of HCHP-nets	
				HSPICE	PowerMill
C432	7759	42.3	0.83	9348	51
C449	7795	39.6	0.92	8472	43
C880	10630	49.3	1.37	7759	36
C1355	14424	90.2	1.92	7512	47
C1908	31135	222.1	3.47	8972	64
C2670	51381	224.9	5.77	8904	39
C3540	75157	577.5	7.13	10540	81
C5315	87132	416.8	10.42	8361	40
C6288	175203	1029.5	17.45	10040	59
C7552	170682	904.3	17.39	9814	52
Average				8972	51.2

For HCHP-nets, the delay values were calculated using HSPICE. The input files for HSPICE and PowerMill are SPICE files which were generated using MOSIS 1.0μ standard cell technology. The Design/CPN tool was used to implement the HCHP-nets and estimate the switching activity at the gate-level. The segmentation for the hierarchical Petri net is made such that the number of arcs between segments is kept to a minimum, this has been done to lower the complexity of the hierarchical Petri net. A random set of 10000 vectors were generated to be given as input stimulus to both PowerMill and HCHP-nets. Due to the time constraint placed by HSPICE, it was not possible to simulate a large sequence of 10000 vectors. Hence, a smaller sequence of 100 vectors was used to calculate the time taken per vector for all the three models. The results of the proposed methodology has been compared with HSPICE and PowerMill. Table 1 describes the ISCAS '85 circuits.

Table 2. PowerMill and HCHP-nets performance for different sequence sizes

Circuit	Time (in sec) for different Input Vector Sequence Sizes			
	4000	6000	8000	10000
C432	3.315	4.959	6.634	8.307
C449	3.683	5.501	7.297	9.270
C880	5.448	8.150	10.911	13.772
C1355	7.724	11.495	15.216	19.201
C1908	13.855	20.243	27.703	34.781
C2670	23.046	34.621	46.005	57.725
C3540	28.520	42.555	57.022	71.353
C5315	41.613	62.423	83.005	104.297
C6288	69.098	104.579	139.183	174.564
C7552	69.557	104.287	139.024	173.914

The simulations were performed on a single processor Sun Ultra 2 workstation with 128 M RAM and a 200 MHz processor. In Table 1, the performance of the HCHP-net simulations are compared with that of HSPICE and PowerMill simulations, based on the per-pattern simulation time. The works presented in [1, 9, 14, 13, 20], provide only power estimates and since our work yields only switching

activity values, it is not possible to compare the results. The work reported in [3] gives comparison only as a relative metric and thus cannot be compared. From the table it can be seen that the HCHP-nets have an average speed-up of 51% over PowerMill. The switching activity values produced by the HCHP-net simulator were exactly same as the ones produced by the other two methods and hence are not listed. It should be noted that the interconnect delays were neglected in this work and the delays were taken from HSPICE. Table 2 gives the time required for simulating HCHP-nets for different sizes of vector sequences.

7 Conclusions

This paper introduces a real-delay simulator for switching activity estimation at the gate-level based on a fast simulation analysis using Hierarchical Colored Hardware Petri nets. The modeling is exact and captures all the correlations in the circuit. In this work, we have assumed that the interconnect delays to be zero which is true for sub-micron designs, but for deep sub-micron designs, this model will fail completely. The future focus of this work is towards developing a complete gate-level real-delay switching activity estimation tool including both interconnect and gate delays. We also propose to extend the model to sequential circuits.

References

[1] Benini L., Favalli M., Olivo P., and Ricco B. A Novel Approach to Cost Effective Estimate of Power Dissipation in CMOS IC's. In *Proc. of the European Design Automation and Test Conf.*, pages 354–360, 1993.

[2] Bhanja S. and Ranganathan N. Dependency preserving probabilistic modeling of switching activity using Bayesian networks. In *Proc. of Design Automation Conf.*, pages 209–214, 2001.

[3] Buhler M., Papesch M., Kapp K., and Baitinger U. G. Efficient Switching Activity Simulation Under a Real Delay Model Using a Bit-Parallel Approach. In *Proc. of the European Design Automation and Test Conf.*, pages 459–463, 1999.

[4] Christensen S., Jorgensen J.B., and Kristensen L.M. Design/"CPN" – "A" Computer Tool for Colored Petri Nets. In E. Brinksma, editor, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 209–223. Springer Verlag, LNCS 1217, 1997.

[5] Ghosh A. A, Devadas S., Keutzer K., and White J. Estimation of Average Switching Activity in Combinational and Sequential Circuits. In *Proc. of the Design Automation Conf.*, pages 253–259, Jun. 1992.

[6] Huber P., Jensen K., and Shapiro R. M. Hierarchies in Colored Petri Nets. In *Advances in Petri Nets, ed. G. Rozenberg,*.

[7] Jensen K. An Introduction to the Theoretical Aspects of Colored Petri Nets. In *Advances in Petri Nets, ed. J. W. Bakker; W.-P. de Roever; G. Rozenberg,*.

[8] Jensen K. *Colored Petri Nets: Basic Concepts*, volume 1. Springer, second edition, 1996.

[9] Jochens G., Kruse L., Nebel W. Application of Toggle-Based Power Estimation to Module Characterization. In *Proc. of PATMOS (Power and Timing Modeling of Integrated Circuits)*, pages 161–170, 1997.

[10] Krishna V., Chandramouli R. and Ranganathan N. Computation of lower and upper bounds for switching activiy using Decision Theory. *IEEE Trans. on VLSI Systems*, 7(1):125–129, Mar 1999.

[11] Meta Software Corporation. *Design/CPN Reference Manual*, 2 edition, 1993.

[12] Najm F. N., Burch R., Yang P., and Hajj I. Probabilistic Simulation for Reliability Analysis of CMOS VLSI Circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):439–450, Apr. 1990.

[13] Rabe D., Jochens G., Kruse L., Nebel W. Power-"Simulation" of Cell based ASIC's: Accuracy-and Performance Trade-Offs. In *Proc. of the European Design Automation and Test Conf.*, pages 356 –36, 1998.

[14] Rabe D., Nebel W. A New Approach in Gate-Level Glitch Modeling. In *Proc. of Design Automation and Test Conf. with EURO-VHDL*, pages 66–71, 1996.

[15] Rokyta P., Fengler W., and Hummel T. Electronic System Design Automation Using High Level Petri Nets. In *Hardware Design and Petri Nets, ed. A. Yakovlev, L. Gomes and L. Lavagno*, pages 193– 204. Kluwer Academic Publishers, 2000.

[16] Roy K., and Prasad S. SYCLOP: Synthesis of CMOS Logic for Low Power Application. In *Proc. of the Intl. Conf. on Computer Design*, pages 464–467, Oct. 1992.

[17] Shen A. A., Ghosh A., Devadas S., and Keutzer K. On Average Power Dissipation and Random Pattern Testability of CMOS Combinational Logic Networks. In *Proc. of the IEEE Intl. Conf. on Computer Aided-Design*, pages 402–407, 1992.

[18] Tan-Li C. and Roy K. Statistical Estimation of Combinational and Sequential CMOS Digital Circuit Activity Considering Uncertainty of Gate Delays. In *Proc. of the ASP-Design Automation Conf.*, pages 95–100, 1997.

[19] Theoharis S., Theodoridis G., and Goutis C. Accurate Data Path Models for Fast RT-Level Power Estimation. In *IEE Proc. Computers and Digital Techniques*, pages 209 –214, Jul. 2000.

[20] Tjarnstrom R. Power Dissipation Estimate by Switch Level Simulation. In *Proc. of the Intl. Symp. on Circuits and Systems*, pages 1157–1161, 1989.

[21] Tsui C. Y., Pedram M., and Despain A. Technology Decomposition and Mapping Targeting Low Power Dissipation. In *Proc. of the Design Automation Conf.*, pages 68–73, Jun. 1993.

[22] Tsui C-Y., Pedram M., Despain A. M. Efficient Estimation of Dynamic Power Consumption under a Real Delay Model. In *Proc. of the Intl. Conf. on Computer Aided Design*, pages 224–228, 1993.