ISCAS 2000 - IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland

Improving Simulation Efficiency for Circuit-Level Power Estimation

Radu Marculescu, Cristinel Ababei

Department of Electrical & Computer Engineering
University of Minnesota, Minneapolis MN 55455
e-mail: {radu,ababei}@ece.umn.edu

Abstract - *In this paper we present an effective technique for compacting a large sequence of input vectors into a much shorter one so as to reduce the circuit-level simulation time by orders of magnitude and maintain the accuracy of the power estimates. In particular, we model the effects of complex spatiotemporal correlations and rise/fall time slopes on total power dissipation. As the results demonstrate, large compaction ratios of orders of magnitude can be obtained without significant loss (about 5%, on average) in the accuracy of power estimates.*

Keywords: *simulation efficiency, power estimation, input dependencies, rise/fall time, vector compaction, Markov models.*

I. INTRODUCTION

Power dissipation has become an important design constraint for nowadays digital systems. Due to this trend, a significant amount of work has been devoted to accurate power estimation in CMOS circuits. To date, both *dynamic* and *static* approaches have been considered, each one having its own advantages and limitations [8]. More precisely, the general simulation techniques provide sufficient accuracy, but are very costly. On the other hand, nonsimulative approaches are much faster, but less accurate than those based on simulation. The reason for inaccuracy is the set of simplifying assumptions (zero rise/fall times, zero-delay) used in calculations.

Another important issue is the *level of abstraction* where the power estimation techniques are applied. Generally speaking, the logic-level power estimation is far less accurate compared to the circuit-level estimation. The circuit-level simulation can improve the accuracy of power estimates but, as a side effect, significantly increase the computational cost of the estimation process.

As a conclusion, a number of issues appear to be important for accurate power estimation. The *input statistics* (spatiotemporal correlations, rise/fall time slopes, etc.) which must be properly captured and the *length of the input sequences* which must be applied are two such issues. Generating a minimal-length sequence of input vectors that satisfies these statistics is not trivial. The reason is the elaborate set of input statistics that must be preserved/ reproduced during sequence generation for power simulators.

The present paper addresses the problem of *circuit-level power estimation* and improves the-state-of the art by providing an original solution under the paradigm of *vector compaction*. Having an initial sequence (representative for a target circuit), we target *lossy compression* [2], i.e. the process of transforming an input sequence into a shorter one, such that the new body of data represents a *good approximation* as far as total power consumption is concerned.

The foundation of our approach is probabilistic in nature; it relies on *adaptive (dynamic) modeling* of PieceWise Linear (PWL) input sequences as second-order Markov sources of information. The adaptive modeling technique itself (a.k.a. *Dynamic Markov Chain* or *DMC modeling* [4]) was introduced recently in the literature on data compression and extended in [9] to handle gate-level spatiotemporal correlations. In this paper, we change the focus from gate-level to circuit-level power estimation and give a new formulation to the vector compaction problem. Using this new formalism, we are able to handle multiple symbols that are used to represent the PWL waveforms that arise in real *Spice* simulations. By moving the vector compaction problem from logic domain to circuit-level, we significantly improve on the accuracy of power estimates and then completely eliminate the limitations in accuracy of power estimates that are typical for logic-level approaches.

As demonstrated by practical evidence, this new framework is extremely effective in power estimation. The basic idea is illustrated in Fig.1a. To evaluate the total power consumption for a PWL input sequence (of length L_0), we first derive the Markov model of the input sequence and then, having this compact representation, we generate a much shorter sequence (of length $L \ll L_0$), equivalent with the initial one, which can be used with any available circuit-level simulator to derive accurate power estimates (Fig.1b).

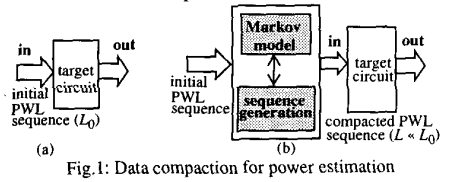

Fig.1: Data compaction for power estimation

We note that, as opposed to current circuit-level techniques that try to improve on the *circuit* model used for simulation, our technique is focusing only on modeling the *PWL sequences* that are fed to the target circuit (simulator). This is a fundamental change which makes our technique appealing in practice: by targeting only the input sequence and not the circuit itself, the approach becomes independent from the circuit modeling part. We also point out that our technique *does not* compete with other circuit-level techniques based on macromodeling [8]; we simply provide a tool for those approaches not only to speedup the characterization process but also to improve on their accuracy by capturing actual spatiotemporal correlations and rise/fall time slopes.

To conclude, both static and dynamic techniques for power estimation may benefit from this research. The issues brought into attention in this paper are new and represent an important step toward reducing the gap between the static and dynamic techniques commonly used in circuit-level power estimation.

The paper is organized as follows: Section II reviews the basic concepts of Markov modeling technique. Section III formalizes, at circuit level, the power-oriented vector compaction problem. In sections IV, we give some experimental results. Finally, we conclude by summarizing our main contribution.

II. BACKGROUND ON DYNAMIC MARKOV MODELS

The foundation of our approach relies on the adaptive (dynamic) modeling of binary input streams as Markov sources of information. The Markov modeling technique [4] was recently extended to capture not only correlations among adjacent bits, but also correlations between successive input patterns [9]. Indeed, for power estimation purposes, this is an essential feature because the power consumption is very much dependent on the statistical properties of the input patterns. In the remaining part of this section, we briefly review the main issues in Markov modeling. For an in-depth presentation the reader is referred to [9].

Without loss of generality, we restrict our attention to finite binary strings; that is, finite sequences on b bits consisting only of 0's and 1's. A particular sequence S_1 consists of vectors $v_1, v_2, \ldots, v_n$ (distinct or not), each having a non-zero occurrence probability. Indices 1, 2,..., n represent the discrete time steps when a particular vector is applied to a target circuit. Imposing a total ordering among bits, such a sequence may be conveniently viewed as a binary tree (called DMT_0 from *Dynamic Markov Tree of order zero*) where nodes at level j correspond to bit j ($1 \le j \le b$) in the original sequence; each edge that emerges from a node is labelled with a positive count (and then a positive probability) that indicates how many times the substring from the root to that particular node, occurred in the original sequence. The DMT_0 alone cannot capture temporal correlations because the relative order of vectors in the initial sequence is irrelevant for its construction. For power estimation this is a fundamental limitation so we consider a more refined structure by incorporating *first-order temporal effects* (called DMT_1 from *Dynamic Markov Tree of order* 1) [9].

Example 1: For the following 4-bit sequence consisting of 8 non-distinct vectors (v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8) = (0000, 0001, 1001, 1100, 1001, 1100, 1001, 1100), the tree DMT_1 is given in Fig.2.

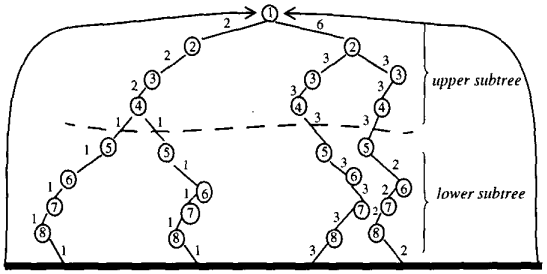

Fig.2: Structure of DMT_1

The upper subtree (levels 1 to 4) represents DMT_0 while the lower subtrees (levels 5 to 8), give the actual sequencing between any two successive vectors. We note that any binary sequence can be modeled as a first-order Markov source using DMT_1. The simple structure of DMT_1, can be further extended to capture temporal dependencies of higher orders. For instance, if we define recursively DMT_2, we can capture second-order temporal correlations. For any sequence where v_i, v_j, v_l are 3 consecutive vectors ($v_i \rightarrow v_j \rightarrow v_l$), the tree DMT_2 looks like in Fig.3.

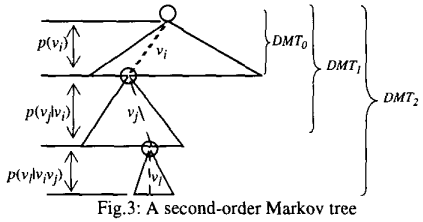

Fig.3: A second-order Markov tree

More generally, a structure DMT_p can be constructed; DMT_p completely captures spatial and temporal correlations of order p [9].

III. Power-Oriented Data Compaction

A. Problem Formulation

Input pattern dependencies (i.e. spatial and temporal correlations) have a dramatic impact on power dissipation estimates [10]. More than this, it was shown that the real rise/fall time slopes in signal propagation can have a significant effect on total power dissipation [3][5][7]. More precisely, different rise/fall time slopes in signal propagation can generate glitches within the circuit and these glitches determine extra-power consumption. In addition, as the signal rise and fall times increase, the contribution of short-circuit power consumption can also significantly increase [1][3]. To illustrate this issue, we simulate the benchmark *C17*, with *Spice* at a clock frequency of 10MHz. The circuit is fed with a 5-bit counted sequence whose signals have, in a first scenario, equal rise and fall times of 1ns. In a second experiment, we change the rise and fall times to 5ns. The total power consumption we obtain is 13.05 μW in the first case and 18.12 μW for the latter, which shows a relative difference of 40%.

Considering all these, we focus on the *input problem*, in the sense that we try to find, *independently* of the target circuit, a good approximation of the input sequence. For circuit-level power estimation purposes, it is critical to distinguish not only between input sequences with very different spatial and temporal correlations, but also different values for the rise and fall time slopes; this is especially true if we are interested in node-by-node accurate power estimation using *Spice*.

To simplify the vector compaction problem, we restrict to a finite set of rise and fall time slopes that may characterize the 0/1 transitions in real applications. More precisely, we consider the set of $2k$ rise/fall time slopes[1] $\{s_1, s_2, \ldots, s_{2k}\}$ that characterize the set of PWL input signals in a *Spice* simulation. We assume that the $0 \rightarrow 1/1 \rightarrow 0$ transitions can take place as shown in Fig.4a. (t_n, t_{n+1}, t_{n+2} represent three discrete time steps associated to the lag-2 Markov chain that characterizes the transition process.)

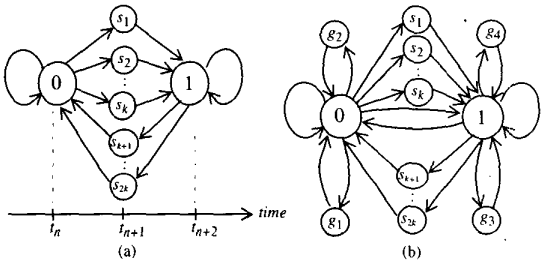

Fig.4: Transitions $0 \rightarrow 1$ / $1 \rightarrow 0$ with k rise and k fall time slopes

In this representation, every transition $0 \rightarrow 1/1 \rightarrow 0$ takes place with a particular slope s_i ($i = 1..k$); moreover, the glitches that have the width below a certain threshold are automatically filtered out. We note that, this modeling scheme can be easily extended to also accommodate $0 \rightarrow 1/1 \rightarrow 0$ transitions with infinite slopes, narrow glitches, and incomplete transitions (symbols g_1, g_2, g_3, g_4 in Fig.4b). However, we restrict our subsequent presentation only to the case illustrated in Fig.4a.

The two-step transition process in Fig.4a can be described using the $(2k+2)$ by $(2k+2)$ transition matrix in equation (1). In this representation, a probability "0" denotes an impossible transition while an entry "*" represents a valid transition (with the probability value within the [0,1] interval). For instance, starting with symbol "0" it's possible either to go to any symbol within the set $\{s_1, s_2, \ldots, s_k\}$ or to remain in the same state symbolized by "0". On the other hand, for any s_i ($i = 1..k$), the process will move deterministically to symbol '1'. We also note that, because Q is a

[1] $s_1, s_2, \ldots, s_k$ are rise-time symbols while $s_{k+1}, s_{k+2}, \ldots, s_{2k}$ are fall-time symbols.

stochastic matrix, the elements in every row add to 1.

$$Q = \begin{array}{c|cccccc} t_n \backslash t_{n+1} & s_1 & \cdots & s_k & s_{k+1} & \cdots & s_{2k} & 0 & 1 \\ \hline s_1 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 1 \\ s_2 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 1 \\ \cdots & & & & & & & & \\ s_k & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 \\ s_{k+1} & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & 0 \\ \cdots & & & & & & & & \\ s_{2k} & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & * & \cdots & * & 0 & \cdots & 0 & * & 0 \\ 1 & 0 & \cdots & 0 & * & \cdots & * & 0 & * \end{array} \quad (1)$$

Using this compact notation, the vector compaction problem can be formulated as follows: for any PWL sequence of length L_0 (consisting of vectors $\alpha_1, \alpha_2, \ldots, \alpha_n$), find another PWL sequence of length $L < L_0$ (consisting of the subset $\beta_1, \beta_2, \ldots, \beta_m$ of the initial sequence), such that the *average transition probability* on the primary inputs of the target circuit is preserved *wordwise.*

More formally, for any generic input value i (that is, 0, 1, or any slope $s_1, s_2, \ldots, s_{2k}$), the following condition should hold:

$$\left\| Q^2_{*|i} P_i - (Q^2_{*|i} P_i)' \right\| = O(\varepsilon) \quad (2)$$

where P_i $(P_i)'$ is the (column) state probability vector of the input value i, $Q^2_{*|i}$ and $(Q^2_{*|i})'$ represent the two-step transition matrices that correspond to the original and compacted sequences, respectively. $O(\varepsilon)$ (read as *'zero of epsilon'*) represents any power series $k_1\varepsilon + k_2\varepsilon^2 + \ldots$ (convergent for small ε).

We note that the above two-step transition probabilities are needed to ensure that we preserve the appropriate (valid) set of triplets (that is, $0 \rightarrow s_1 \rightarrow 1$ or $0 \rightarrow s_2 \rightarrow 1$, etc.) which define the non-ideal (valid) $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions. Furthermore, this bit-level formulation can be extended to any number of bits.

B. A DMC-based Approach

The compaction process of the PWL sequences can be realized using the DMC model in Section II. To this end, all we need to do is to *encode* the PWL information in *binary format* and then apply the DMC compaction procedure at bit-level. For example, for a PWL sequence that contains symbols from the set $\{0,1,s_1,s_2,s_3,s_4\}$ (i.e. 0, 1, two rise, and two fall time slopes), one can use a 3-bit encoding to generate the binary sequence. The overall flow is given in Fig.5.

Starting with a real *Spice* sequence, we first linearize it by considering the full transitions between 0 and 1 with different rise/fall time slopes. After that, using an encoding scheme with $\lfloor \log(2k+2) \rfloor$ bits, we preprocess the PWL sequence (of length L_0) into a binary one (S_0 in Fig.5) which is taken as input in the compaction module. The resulting compacted sequence (i.e. S_0') is decoded and transformed back into a PWL sequence (of length L) which is used as stimulus at the primary inputs of the target circuit.

The compaction process in Fig.5 is based on the second-order dynamic Markov tree (DMT_2 in Fig.3) which actually implements relation (2) at word-level. A practical procedure to construct the DMT_2 and generate the compacted sequence works as follows: during a one-pass traversal of the original sequence (when we extract the bit-level statistics of each individual vector $v_1, v_2, \ldots, v_n$ and also those statistics that correspond to pairs of consecutive vectors (v_1v_2), (v_2v_3),..., $(v_{n-2}v_{n-1})$, $(v_{n-1}v_n)$), we grow simultaneously the tree DMT_2. We continue to grow DMT_2 as long as the number of nodes in the Markov model is smaller than a user-specified threshold and we didn't reach the end of the sequence. If the Markov model becomes too large, we just generate the new sequence up to that point and discard (flush) the model. A new Markov model is started again and the process is continued up to the end of the original sequence.

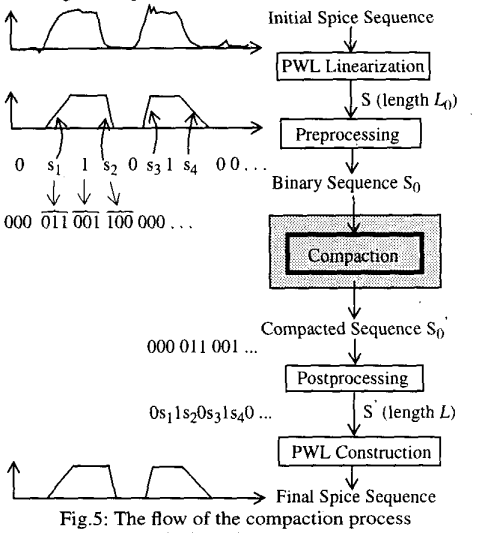

Fig.5: The flow of the compaction process

Once the Markov model is built, to generate a new sequence, we use a modified version of the *dynamic weighted selection algorithm* [6]. To ensure a minimal level of error, we implemented an *error controlling mechanism* in a greedy fashion. More precisely, at each level in the lower Markov tree, to decide whether a zero or one has to be generated, we compute the transition probabilities for both alternatives and choose the one that minimizes the absolute error accumulated up to that point. Simultaneously, the upper tree is parsed from the root to the leaves, according to the bits generated in the lower subtree. The procedure is then resumed until the needed number of vectors is generated.

This strategy does note introduce 'forbidden' vectors that is, those input patterns that did not occur in the original sequence, will not appear in the compacted sequence either. This is an essential capability to avoid 'hang-up' ('forbidden') states of the circuit during simulation process for power estimation. We also note that, by construction, it does not introduce forbidden transitions either.

IV. Experimental Results

The overall strategy is given in Fig.6. In our current implementation, the input data is obtained from real *Spice* simulations by doing piecewise linearization and representation of the resulting waveforms with six symbols (i.e. 0, 1, and s_1, s_2, s_3, s_4 that correspond to two different rise and two different fall slopes). Then, encoding the PWL of length L_0 (represented with symbols in $\{0,1, s_1,s_2,s_3,s_4\}$) with 3 bits, we transform the original input stream into a binary input sequence which is sent to the compaction module. Starting with this binary input sequence, we perform a one-pass traversal of the original sequence and simultaneously build the basic tree DMT_2.

Table 1: Power estimates and simulation times for the original and compacted sequences

			Initial Sequences (non-zero rise/fall times)		Ideal Sequences (zero rise/fall times)		Compacted Sequences (r = 5) (non-zero rise/fall times)			Compacted Sequences (r = 10) (non-zero rise/fall times)		
Circuit	Inputs	Gates	Power [μW]	time [sec]	Power [μW]	Diff.[%]	Power [μW]	Error [%]	time [sec]	Power [μW]	Error [%]	time [sec]
cu	14	44	34.36	110.52	25.26	26.48	37.12	8.03	20.29	36.84	7.21	9.78
add16	33	80	280.78	944.01	211.20	24.78	277.30	1.23	179.42	261.89	6.70	80.77
rd84	8	125	257.89	817.50	205.08	20.47	241.06	6.52	116.32	227.75	11.68	50.70
9symml	9	145	219.13	971.13	191.34	12.68	230.17	5.03	192.16	212.61	2.97	82.55
my_adder	33	173	311.02	978.77	284.28	8.59	318.97	2.55	185.98	323.83	4.11	86.84
mul4	16	189	337.65	2030.15	284.88	15.62	341.66	1.18	381.36	326.87	3.19	179.11
alu2	10	277	319.71	1919.37	293.18	8.29	317.64	0.64	397.74	305.83	4.34	176.81
C1355	41	292	557.16	4017.79	446.05	19.94	568.38	2.01	789.36	534.03	4.15	352.38
t481	16	327	380.81	2071.00	362.12	4.90	362.56	4.79	355.75	359.04	5.71	163.31
C499	41	352	636.61	4123.90	524.69	17.58	645.03	1.32	879.76	613.48	3.63	372.68
C1908	33	384	577.25	4352.41	498.45	13.65	598.16	3.68	816.54	574.39	0.49	420.72
alu4	14	561	690.16	5752.30	651.33	5.60	698.71	1.23	1085.58	688.53	0.23	490.06
mul8	32	820	1039.70	10755.00	912.43	12.24	1106.40	6.41	2373.08	1039.40	0.00	992.51
					Avg. values	Diff. = 14.67%		Err. = 3.43%	Speedup = 5.29		Err. = 4.18%	Speedup = 11.86

The next step in Fig.6 does the actual generation of the output sequence. Once we get the compacted binary sequence, we generate a PWL sequence (of length L) within the postprocessing module. This sequence is later used for *Spice* simulations. If the initial PWL sequence has the length L_0 and the new generated sequence has the length $L < L_0$, then a *compaction ratio* of $r = L_0/L$ is achieved.

Finally, a validation step is included in the strategy; we simulate the target circuit with the original *Spice* sequence and the one resulted from the compaction process and then we compare the results in terms of average power consumption estimates.

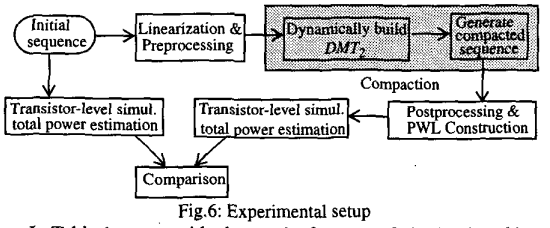

Fig.6: Experimental setup

In Table 1, we provide the results for a set of highly biased input sequences (of length 850 vectors) obtained from real applications. These sequences are compacted with two different compaction ratios (r = 5 and 10). We give in this table the total power dissipation measured for the initial (column 4) and compacted sequences (columns 8, 11). For comparison purposes, we also indicate (columns 6 and 7) the power and the corresponding difference we get if we ignore the actual rise and fall times of the input signals. In columns 5, 10 and 13, we give the time in seconds (on an Ultra 10 workstation with 128 Mbytes of memory) necessary to estimate the average power consumption for the initial and compacted sequences, respectively. Since the compaction process with DMC modeling is linear in the number of nodes in the structure of the DMT_2, the time needed for compaction is less than 3 seconds in all cases. During these experiments, the number of states allowed in the Markov model was 5,000 (about 140 Kbytes).

As we can see, the quality of results is very good even when the length of the initial sequence is reduced by one order of magnitude. For instance, the *Spice* simulation of *alu2* took 1919.37 sec to estimate an exact power value of 319.71 μW, whereas using the compacted sequence of only 170 vectors (r = 5), *Spice* estimated a value of 317.64 μW in only 397.74 sec. Furthermore, with only 85 vectors (r = 10), *Spice* estimated a power consumption of 305.83 μW in only 176.81 sec. This reduction in the sequence length has a significant impact on speeding-up the simulative approaches where the run time is proportional to the length of the sequence which must be simulated. As it can be seen in Table 1, for r = 10, the average relative error is less than 5%, while the speed-up in power estimation is more than one order of magnitude, on average.

V. Conclusion

In this paper, we addressed the circuit-level power estimation under the paradigm of vector compaction. Based on dynamic Markov modeling, we proposed a new approach to compact an initial PWL sequence into a much shorter equivalent one, which can be used with any available simulator to obtain accurate power estimates.

The results obtained on common benchmarks show that large compaction ratios can be obtained without much loss in the accuracy of power estimates. We are currently extending our approach to the more general case described in Section III.

References

[1] V. Adler, E.G. Friedman, 'Delay and Power Expressions for a CMOS Inverter Driving a Resistive-Capacitive Load', *Analog Integrated Circuits and Signal Processing*, 14, 29-39, Kluwer Academic Publishers, 1997.

[2] T. Bell, J. Cleary, I. Witten, 'Text Compression', Prentice Hall, 1990.

[3] L. Bisdounis, S. Nikolaidis, 'Propagation Delay and Short-Circuit Power Dissipation Modeling of the CMOS Inverter', *IEEE Trans. on Circuits and Systems-I*, vol. 45, No. 3, pp. 259-270, March 1998.

[4] G. V. Cormack, R.N. Horspool, 'Data Compression Using Dynamic Markov Modeling', *in Computer Journal*, Vol. 30, No. 6, pp. 541-550, 1987.

[5] M. Favalli, L. Benini, 'Analysis of Glitch Power Dissipation in CMOS ICs', *Proc. Intl. Symp. Low-Power Electronic Design (ISLPED)*, 1995.

[6] J. W. Green, K. J. Supowit, 'Simulated Annealing without Rejected Moves', in *Digest. of Intl. Conference on Computer Design*, pp. 658-663, *Oct. 1984.*

[7] Y. I. Ismail, E. G. Friedman, J. L. Neves, 'Dynamic and Short Circuit Power of CMOS Gates Driving Losseless Transmission Lines', *IEEE Trans. on Circuits and Systems-I*, vol. 46, No. 8, pp. 950-961, March 1999.

[8] E. Macii, M. Pedram, F. Somenzi, 'High level power modeling, estimation and optimization', *IEEE Trans. on CAD of ICS*, Vol. 17. No. 11, Nov. 1998.

[9] R. Marculescu, D. Marculescu, M. Pedram, 'Sequence Compaction for Power Estimation: Theory and Practice', *IEEE Trans. on CAD of ICS*, vol. 18, No. 7, pp. 973-993, July 1999.

[10] R. Marculescu, D. Marculescu, M. Pedram, 'Probabilitic Modeling of Dependencies During Switching Activity Analysis', *IEEE Trans. on CAD of ICS*, vol. 17, No. 2, pp. 73-83, Feb.1998.