A Probabilistic Approach for RT-Level Power Modeling

José Costa, José Monteiro, L. Miguel Silveira
{jccc,jcm,lms}@algos.inesc.pt
IST/INESC, Lisboa, Portugal

Srinivas Devadas
devadas@caa.lcs.mit.edu
MIT, Cambridge, MA, USA

We propose a method for register-transfer level (RTL) power modeling. The switched capacitance and switching probability of each output of a functional module are modeled by formulas that are a function of the module's inputs probabilities. These formulas are computed beforehand for each module using the polynomial simulation scheme, and stored in the module library. The switched capacitance (and thus power) for each instance of a module in the circuit can then be efficiently evaluated for its specific input probabilities. The switching probabilities at the outputs of each module can be computed in a similar manner, thus providing a means of propagating the switching probabilities through the circuit described at the RT level.

We provide a set of experimental results that show that this method provides estimates close to the logic-level estimates, but is orders of magnitude faster.

I. Introduction

Power consumption has become a major concern in the design of integrated circuits. Two independent factors have contributed for this. On one hand, low power consumption is essential to achieve longer autonomy for portable devices. On the other hand, increasingly higher circuit density and higher clock frequencies are creating heat dissipation problems, which in turn raise reliability concerns and lead to more expensive packaging.

In the last few years, research on techniques for low power at various levels of design has intensified. Initial work was done at the logic and circuit levels. However, much research is now being conducted at the register-transfer and behavioral levels as it has been recognized that decisions taken during high-level synthesis can have a much larger impact on the power dissipation of the final circuit. Hence, the need for efficient and accurate power analysis tools at high levels of abstraction has arisen.

In register-transfer-level (RTL) power analysis, one wishes to estimate the power consumed by a logic implementation given only a RTL description of circuit behavior. It is assumed that the RTL description contains enough information to synthesize a functionally equivalent logic implementation. This implementation can be significantly more complex than the RTL description. It is not unusual for a RTL description consisting of a few tens of lines to result in a logic circuit comprised of thousands of gates; the main reason for this is that library modules such as adders and multipliers are merely represented as $+$'s and $*$'s at the RT level, but are expanded at the logic level. Given this increase in complexity, it is highly desirable to develop RTL power estimation algorithms.

In this paper, we describe a probabilistic approach for the power analysis of circuits described at the RT level. The switched capacitance and switching probability of each output of a functional module are modeled by formulas that are a function of the module's inputs probabilities. These formulas are computed beforehand for each module and stored in the module library. We present a *polynomial simulation* method that we use for this library module characterization process. This method is a generalization of the exact signal probability evaluation algorithm due to Parker and McCluskey [10] and handles arbitrary transport delays.

The switched capacitance (and thus power) for each instance of a module in the circuit can then be efficiently evaluated for its specific input probabilities. The switching probabilities at the outputs of each module can be computed in a similar manner, thus providing a means of propagating the switching probabilities through the circuit described at the RT level.

In Section II, we discuss previous work on power analysis techniques at the RT level. We describe the polynomial simulation method and the RTL power modeling method that we are proposing in Section III. In Section IV, we provide a set of experimental results that show that the models obtained for functional modules provide very good accuracy. We present some conclusions and discuss future work in Section V.

II. Previous Work on Power Estimation at the RTL

At the register-transfer level (RTL), the circuit is described in terms of interconnected functional modules of varying degrees of complexity. The general approach to compute the power dissipation of a logic circuit described at the RT level is to estimate the power of each of these modules separately. Even if the details of the specific implementation of a module are known, for the sake of efficiency a much simpler model for the module is used.

A. Simulation-based Methods

The basic model used by most power estimation techniques for a functional module is of the form [11]:

$$P = C_{\mathit{eff}} K V_{DD}^2 f \tag{1}$$

where V_{DD} is the voltage of the power supply for the module and f is the clock frequency at which it operates. K is the scaling factor, as a general assumption is that power scales with area. For instance, if N is the word length, $K = N$ for an adder and $K = N^2$ for a multiplier. Lastly, the factor C_{eff} is the average switched capacitance per bit slice of the functional unit. C_{eff} is computed beforehand (typically using random simulation) for each type of functional module and stored in the module library.

This power estimation model for functional modules is extended in [6] in two ways. First, the authors note that it may require more than a single parameter to describe how the switched capacitance varies with the input width for a given type of module. For example, a register file with R registers and word length N should be described by capacitance values that scale with R to model the control logic, a function of how many words can be stored, N for the input/output buffers, and RN, the number of storage cells. The switched capacitance is then given by

$$C_0 R + C_1 N + C_2 NR = \boldsymbol{C}_{\mathit{eff}}^T \cdot \boldsymbol{K}$$

where $\boldsymbol{C}_{\mathit{eff}} = [C_0 \;\; C_1 \;\; C_2]^T$ and $\boldsymbol{K} = [R \;\; N \;\; RN]^T$.

A second observation made in [6] is that assuming a uniform distribution for all the inputs to the module can be very inaccurate. In a general datapath circuit, independently of the correlation between consecutive input vectors, the least significant bits present a uniform probability of switching. On the other hand, the switching probability of the most significant bits is highly dependent on the correlation value.

The previous approaches can model the glitching activity that is generated within a module by taking it into account during the characterization phase of the module. However, they assume that the inputs to the modules are glitch free. A different power model is proposed in [12] that is able to handle glitching activity at the inputs.

B. Probabilistic Methods

The methods described above are all simulation-based. C_{eff} in Eqn. (1) and all the coefficients used in [12] are obtained by simulating each different module using a large number of input vectors, randomly generated using a uniform distribution. Since exhaustive simulation is not possible in practice, the accuracy of these methods is not only limited by the simplicity of the model, but also by the number of simulation runs used in the characterization step.

A more severe limitation is that this characterization process is done using fixed statistics for the inputs (generally, uniform distribution is assumed) and may breakdown when the actual input statistics for the instance of a module differ from the values assumed when the power expression was computed. Probabilistic methods do not suffer from these shortcomings. However, the complexity of these methods can be high and thus approximations may have to be used.

Two probabilistic approaches for power estimation at the RT level were proposed in [9] and [7]. They are both based on the concept of entropy and their focus is to derive implementation-independent measures of the signal activity in the circuit. The first assumption both methods make is that the average power dissipation of the module can be approximated by

$$P_{avg} \propto \sum_{\substack{\text{all gates } g \\ \text{in module}}} C_g N_g \approx \mathcal{CN} \propto \mathcal{CH} \qquad (2)$$

where $\mathcal{C}$ represents the total capacitance, $\mathcal{N}$ the average node switching activity and $\mathcal{H}$ the average entropy in the module. The entropy of a Boolean signal x having a probability p_1 of being 1 follows very closely the switching probability of a signal, $2p_1(1-p_1)$ (ignoring temporal correlation).

A number of assumptions are made in both [9] and [7] on how to propagate the entropy of the inputs through the circuits. These methods can be very efficient, though given all the required approximations and the fact that they ignore issues such as glitching implies that these techniques are not very robust in terms of accuracy.

More recently, the use of ADDs [1] has been proposed to store the switching capacitance of modules [2]. This method takes into account the gate-level structure of the module. However its complexity limits the size of the circuit it can be applied to. Further, the method of [2] does not provide a means of computing the output switching of a module given its input statistics.

In this paper, we present a method for computing an expression that models the switched capacitance of a functional module. For each instantiation of a module its power dissipation can be efficiently computed by evaluating this expression for the specific input probabilities. Additionally, expressions for the probabilities of the outputs of a module are computed, thus enabling the propagation of probabilities through the RTL circuit.

III. Polynomial Simulation

The computation of the expressions that model the switched capacitance and the probability of each output of a module is based on the Parker-McCluskey method [10]. A desirable feature of this method is that spatial correlation between internal signals is accurately taken into account. In this section, we briefly describe this method and extensions to handle temporal correlation and generic delays. We also present the approximation scheme we use to keep the expressions obtained manageable.

A. The Parker-McCluskey Method

The Parker-McCluskey method evaluates signal probabilities of a Boolean function f with inputs $x_1, \ldots, x_N$ by generating a polynomial that represents the probability that the gate output is a 1, for each gate in the circuit. It follows basic rules for propagating polynomials through logic gates.

Definition: Given a polynomial $P(x_1, \ldots, x_N)$, the function supexp(P) is defined as the polynomial resulting from replacing each ${x_i}^k \in P$ with x_i for all $k > 1$.

For example, if $P = {x_1}^2 + x_1 \cdot x_2$, supexp($P$) = $x_1 + x_1 \cdot x_2$.

Given a polynomial P_g for gate g, if g is an input to an inverter, the polynomial for the output of the inverter is $1 - P_g$. Given polynomials P_{g_1} and P_{g_2} at the inputs of an AND gate h, the polynomial for the output of the AND gate will be P_h = supexp($P_{g_1} \cdot P_{g_2}$). For an OR gate, P_h = 1 - supexp($(1-P_{g_1}) \cdot (1-P_{g_2})$) = P_{g_1} + supexp($(1-P_{g_1}) \cdot P_{g_2}$).

We begin with the primary input polynomials x_1 through x_N, and traverse the circuit from inputs to outputs to obtain $P_f(x_1, \ldots, x_N)$, for each gate f in the circuit. Given a probability value for each x_i, namely $pr(x_i)$, $pr(f) = P_f(pr(x_1), \ldots, pr(x_N))$.

B. Transition Probabilities

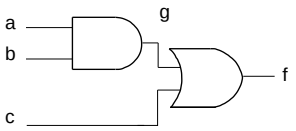
The Parker-McCluskey algorithm can be generalized to work with transition probabilities [3]. Each input x_i has four probability variables associated with it, corresponding to the input staying low, making a rising transition, making a falling transition, and staying high. These are x_i^{00}, x_i^{01}, x_i^{10}, and x_i^{11}, respectively. For each gate g, we now have four polynomials P_g^{00}, P_g^{01}, P_g^{10}, and P_g^{11}, corresponding to the probability that the gate stays low, makes a rising transition, makes a falling transition, and stays high, respectively. We will refer to these four polynomials as the **polynomial group** for a gate. Simulation tables can be used to obtain the basic rules for computing the polynomial group for the output of each type logic gate.

We will always be manipulating polynomial groups henceforth, and for clarity, we will represent the polynomial group $\{P_g^{00}, P_g^{01}, P_g^{10}, P_g^{11}\}$ as P_g.

C. Gate Delay Effects and Polynomial Waveforms

The Parker-McCluskey method has also been extended to handle gate delays [4]. At each gate output we will have a waveform of polynomial groups, termed a **polynomial waveform**, where each group represents the conditions at the gate output at a particular time instant. We denote the polynomial group for gate g at time instant t as $P_g[t]$.

For example, in the simple circuit of Figure 1(a) with unit gate delays, we will have, for the various signals, the polyno-



(a)

$a:$ $P_a[0]$
$b:$ $P_b[0]$
$c:$ $P_c[0]$
$g:$ $P_g[0], P_g[1]$
$f:$ $P_f[0], P_f[1], P_f[2]$

(b)

Fig. 1. Unit-delay example circuit.

mial waveforms given in Figure 1(b) representing the different time instants that each input/gate can make transitions.

We need a polynomial simulation algorithm that can simulate a gate-level network with arbitrary gate delays. Given primary input polynomial waveforms the algorithm should generate polynomial waveforms for each gate output. Such an algorithm is described in pseudo-code in Figure 2.

The simulator processes one gate at a time, moving from the primary inputs to the primary outputs of the circuit. For each gate g_i, an ordered list of the possible transition times of its inputs is first obtained. Then, possible transitions at the output of the gate are derived, taking into account transport delays from each input to the gate output. The processing done is similar to the "time-wheel" of a timing simulator. The output's polynomial group is computed through G_i, the simulation calculus for gate g_i. When the size of the polynomial becomes very large (parameterizable through a user-specified MAX_TERMS), we use an approximation scheme to obtain a simpler polynomial that best approximates the original one. This process is described next.

D. Approximation Scheme

The Parker-McCluskey algorithm cannot be used on large circuits, since it involves "collapsing" the circuit into two levels. We use interpolation and the least squares method [5] to obtain a simpler polynomial that is closest to the original polynomial for a large fraction of the input space.

We first generate some interpolation values by evaluating the original polynomial at several points in the input space. All the input probabilities ($x_i^{00}, x_i^{01}, x_i^{01}, x_i^{01}$, with $1 \leq i \leq n$, where n is the number of primary inputs the polynomial is a function of) are initially set to 0.25. For each primary input i, we evaluate the polynomial at different combinations of the probability values of i and $i+1$ over k selected points. Thus, a polynomial with n inputs will be evaluated at $k^2 n$ different points.

The approximating polynomial we are using is of the form:

$$P' = a_0 + \sum_{i=1}^{n} \left(a_{3i-2} P_i^{00} + a_{3i-1} P_i^{01} + a_{3i} P_i^{10} \right)$$

Note that the term corresponding to P_i^{11} is not in the polynomial as it is completely determined by the other three terms of the same input variable.

We are using the least squares method to compute the a_j's so that P' best approximates the interpolation points. This method involves the solution of a linear system of equations with $3n+1$ unknowns. We used a value of $k = 6$ in our experiments because that gave the best tradeoff between run-time and accuracy. For $k < 6$ the results degrade very rapidly and there is no noticeable improvement by using larger k.

IV. Experimental Results

In this section, we present power estimation results using the methods described in the previous sections. The module

Polynomial_Simulation ($Module$)
1. Init_Polynomial_Waveforms (Primary_Inputs ($Module$)) ;
2. $Gates$ = Topological_Sort($Module$) ;
3. for each g_i in $Gates$ {
4. Δ = delay of g_i ;
5. TP = NIL(LIST) ;
6. for each input g_j of g_i ($gi_1, \cdots, gi_m$) {
7. for each time point (k, $P_{g_j}[k]$) of g_j {
8. TP = InsertInOrder (TP, (k, $P_{g_j}[k]$)) ;
9. }
10. }
11. for each new time point k in TP {
12. $P_{g_i}[k+\Delta] = G_i(P_{gi_1}[k], \cdots, P_{gi_m}[k])$;
13. if(# terms in $P_{g_i}[k+\Delta] >$ MAX_TERMS) {
14. ApproximatePolynomial($P_{g_i}[k+\Delta]$) ;
15. }
16. }
17. }

Fig. 2. Pseudo-code for the polynomial simulation algorithm.

characterization routine has been developed within the SIS [13] framework.

We have applied our module characterization approach based on polynomial simulation to a set of datapath modules. We present results that show that the evaluation of the switched capacitance polynomial for different input probabilities gets very close to the exact value obtained with symbolic simulation [8].

In Table I we present results for a set of 8-bit datapath circuits: carry-lookahead (`add_cla8`), ripple-carry (`add_rpl8`) and carry-bypass (`add_cbp8`) adders, one subtracter (`sub8`), greater-than (`greater8`) and equality (`equal8`) comparators, one multiplexor (`mux8`) and a 4-bit barrel-shifter (`shifter4_8`). These modules were read into SIS and a polynomial representing the switched capacitance of the module as a function of the input probabilities was obtained. Polynomials representing the transition probabilities of the modules outputs were computed in a similar manner. For the results presented, a unit-delay model was used in this characterization process, although zero or general delay models could as easily been used.

The CPU time required for this process is shown in the second column of Table I, obtained on a Sun 5/85 with 64MB of main memory. As it can be observed, this time can be high, due to the need of solving many large linear systems of equations involved in the approximation process. However, it should be stressed again that this operation needs to be done **only once** for each module during the characterization process of the library. The power estimates under column "Poly." (consisting of simply evaluating the power polynomial) were all computed under 1 second of CPU time.

In order to confirm the accuracy of the approximations made using the least squares method, we evaluated the power polynomials with different sets of input probabilities, corresponding to lines a, b and c:

a) all inputs have all probabilities set to 0.25;
b) the probabilities are within 0.1 of 0.25, assigned randomly to each input;
c) the probabilities can have any value, again assigned randomly to each input.

We compare the results obtained with the exact value, computed using symbolic simulation [8]. In Table I we present the power estimates using symbolic simulation for each of the input probabilities under "Exact" and the power estimates ob-

TABLE I
EVALUATION OF THE SWITCHED CAPACITANCE POLYNOMIAL FOR DIFFERENT INPUT PROBABILITIES.

Circuit Name	Char. Time	Prob.	Power		
			Exact	Poly.	err
add_cla8	2.9h	a	376.3	372.3	1.1
		b	382.4	381.4	0.3
		c	318.2	313.2	1.6
add_rpl8	3.0h	a	384.3	388.2	1.0
		b	392.4	397.9	1.4
		c	333.1	396.6	19.1
add_cbp8	2.1h	a	415.2	398.9	3.9
		b	426.5	408.1	4.3
		c	367.2	393.4	7.1
sub8	3.4h	a	536.4	544.5	1.5
		b	536.5	551.8	2.9
		c	395.3	465.1	17.6
shifter4_8	17.1m	a	595.3	594.5	0.1
		b	583.4	575.7	1.3
		c	531.1	526.0	1.0
equal8	14.2m	a	141.4	141.4	0.0
		b	140.7	140.6	0.1
		c	117.0	118.1	0.9
greater8	9.5h	a	224.7	233.6	4.0
		b	222.6	231.7	4.1
		c	181.4	188.0	3.6
mux8	1s	a	180.0	180.0	0.0
		b	178.9	178.9	0.0
		c	153.6	153.6	0.0

tained by evaluating the power polynomial under "Poly.". The power values are in micro-watts, assuming a 5V supply voltage and a 20MHz clock frequency. The last column of Table I shows the percentage error of the estimate using polynomial simulation and the exact estimate.

As we can see from this last column, the errors are acceptable for a wide range of the input space. When the switching probability of the inputs is 0.25, the error is very low. This error increases slightly for case 2 and is more noticeable for case 3. This is due to the fact that most of the points that we are using for interpolation in our approximation scheme (Section III-D) are close to 0.25, therefore the approximate polynomial that we obtain is closer to the exact in this region of the input space. In practice, in datapath circuits, the input probability is many times close to 0.25, thus this approximation may be sufficiently accurate. Still, we will continue to investigate methods that best approximate the exact solution for a larger region of the input space.

Note that for the 8-bit multiplexor the error is always zero since no approximations for the polynomials were required. This also explains why the characterization process is so fast.

V. CONCLUSIONS AND FUTURE WORK

We have presented a probabilistic approach to model the power dissipation of a circuit described at the register-transfer level. Our method is based on polynomial simulation which propagates polynomials representing switching activity through combinational logic and memory elements.

Basic combinational logic modules can be characterized using the polynomial simulation method, where the power dissipation as well as output switching activities are represented as sets of polynomials. Once a module has been characterized, if it is encountered in a RTL circuit, the output activities and power dissipation can be calculated efficiently and accurately for arbitrary input activities. Thus, the entire power dissipation of an RTL circuit can be calculated efficiently using this scheme. Our experimental results indicate a significant speed-up over logic-level estimation with only a small loss of accuracy.

Future work involves more extensive experimentation with the polynomial interpolation, and the development of methods for the direct characterization of sequential modules such as finite state machines.

VI. ACKNOWLEDGEMENTS

This work was supported in part by "Fundação para a Ciência e Tecnologia" under project "Praxis XXI".

REFERENCES

[1] R. Bahar, E. Frohm, C. Gaona, G. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic Decision Diagrams and their Applications. In *Proceedings of the International Conference on Computer-Aided Design*, pages 188–191, November 1993.

[2] A. Bogliolo, L. Benini, and G. De Micheli. Characterization-Free Behavioral Power Modeling. In *Proceedings of the Design Automation and Test in Europe*, pages 767–773, March 1998.

[3] D. Cheng. *Power Estimation of Digital CMOS Circuits and the Application to Logic Synthesis for Low Power*. PhD thesis, University of California at Santa Barbara, December 1995.

[4] J. Costa, J. Monteiro, and S. Devadas. Switching Activity Estimation using Limited Depth Reconvergent Path Analysis. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 184–189, August 1997.

[5] G. Golub and C. Van Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore, Maryland, second edition, 1989.

[6] P. Landman and J. Rabaey. Activity-Sensitive Architectural Power Analysis. *IEEE Transactions on Computer-Aided Design*, 15(6):571–587, June 1996.

[7] D. Marculescu, R. Marculescu, and M. Pedram. Information Theoretic Measures of Energy Consumption at Register Transfer Level. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 81–86, April 1995.

[8] J. Monteiro, S. Devadas, A. Ghosh, K. Keutzer, and J. White. Estimation of Average Switching Activity in Combinational Logic Circuits Using Symbolic Simulation. *IEEE Transactions on Computer-Aided Design*, 16(1):121–127, January 1997.

[9] F. Najm. Towards a High-Level Power Estimation Capability. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 87–92, April 1995.

[10] K. Parker and E. McCluskey. Probabilistic Treatment of General Combinational Networks. *IEEE Transactions on Electronic Computers*, C-24(6):668–670, 1975.

[11] S. Powell and P. Chau. Estimating Power Dissipation of VLSI Signal Processing Chips: the PFA Technique. In *VLSI Signal Processing IV*, pages 250–259, 1990.

[12] A. Raghunathan, S. Dey, and N. Jha. Register-Transfer Level Estimation Techniques for Switching Activity and Power Consumption. In *Proceedings of the International Conference on Computer-Aided Design*, pages 158–165, November 1996.

[13] E. Sentovich et al. *SIS: A System for Sequential Circuit Synthesis*. University of California, Berkeley, April 1992.