LEE: A LEAKAGE ESTIMATION ENVIRONMENT

[1]Mateus V. Gomes, [1]Fábio R. Pereira, [1]Leomar S. Rosa Jr, [2]Paulo Butzen, [2]André I. Reis, [1]Renato P. Ribas

[1]Instituto de Informática – UFRGS, Porto Alegre, Brazil
[2]Nangate Inc., Menlo Park, CA, Herlev, Denmark
{mvngomes, frpereira, leomarjr, rpribas@inf.ufrgs.br}, {pbu, are}@nangate.com

ABSTRACT

Power consumption is currently an important issue in digital circuit design. Minimal leakage dissipation represents a key factor for emerging downscaling transistor technologies. Leakage Estimation Environment (LEE) tool, presented herein, allows fast subthreshold leakage power estimation in digital CMOS circuits, by using logic and probabilistic evaluations of signals through the circuit. The software development and experimental results are discussed in this paper.

1. INTRODUCTION

Aggressive scaling of CMOS devices has resulted in higher integration density and improved performance. Simultaneously, static power consumption has become an important issue due to emergent mobile products. Standby currents are increasing significantly in advanced submicron and sub-100nm technologies, where threshold voltage and gate oxide thickness of transistors tend to reduce. As a consequence, great effort has been concentrated in understanding the leakage mechanisms, modeling their behavior and developing design techniques for static power saving [1]-[6].

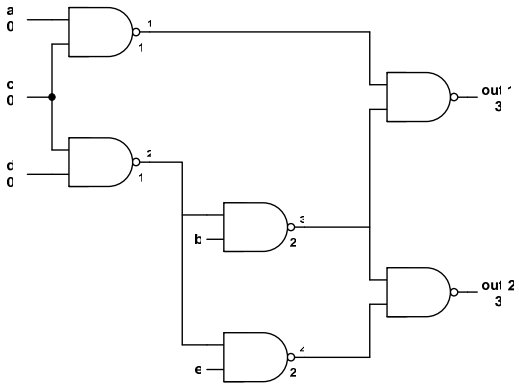
Leakage power estimation is absolutely necessary for designing low-static-power digital circuits. Among different leakage mechanisms, two major ones can be identified: subthreshold leakage and gate oxide leakage. Several leakage prediction methods have been reported in the literature [2]-[6]. Gu *et al.* in [2] and Cheng *et al.* in [3] estimate subthreshold leakage current by using analytical models, while Yang *et al.* in [4] includes the gate leakage component in the analysis. These models are usually computer timing consuming in huge circuits due to the analytical method complexity.

A fast estimation of subthreshold leakage model is presented in [6]. This work predicts subthreshold current based on the device electrical conductance association. Gate leakage current is also fast predicted based on the transistor bias condition, as reported in [5]-[6]. These approaches are not as accurate as complex analytical models, but they provide leakage power values useful to be used as a cost data in technology mapping procedure when low power system is addressed.

This work presents the LEE tool for the circuit-level leakage analysis. Digital circuits, described through Boolean equations of cells and their connectivity, are evaluated according to their functionality in order to determine the logic values and the signal probabilities of internal and output nodes. The analysis of individual cells is provided by specific methods, not treated herein. In other words, the proposed work offers a mechanism useful to evaluate logic cell leakage models at circuit-level.

2. ENVIRONMENT DESCRIPTION

The tool receives as entry data the circuit description in Boolean equation format, as illustrated in Fig. 1. It is then loaded into the data structure, which consists of a set of logic cells, with their respective equations, inputs and outputs, as well as a hash map of the input variables with the corresponding logic values and signal probability.



(a)

```
INORDER = a b c d e;
OUTORDER = out1 out2;
i1 = !(c * a);
i2 = !(d * c);
i3 = !(b * i2);
out1 = !(i3 * i1);
i4 = !(e * i2);
out2 = !(i3 * i4);
```

(b)

Figure 1 – 'C17' ISCAS benchmark circuit description: (a) schematic; (b) equation format.

When the circuit is loaded, all input logic values are set to '0' logic value, and the input data probabilities are set to 0.5 as default value. It is possible to change all of them at a time or only in a specific pin. After setting the input data condition, the following tasks are performed: functional behavior, signal occurrence probability and cell leakage evaluation.

The functional evaluation of the circuit generates the logic values of the internal nodes according to the each logic cell functionality. The probability evaluation, in turn, provides the signal value probability in each circuit node from the primary input definition. Finally, the leakage evaluation applies a certain leakage estimation procedure to each individual cell, according to the cell input condition. Note that, the standby currents are state dependents, i.e., they present different values depending on the input signals.

The Leakage Evaluation Environment – LEE tool was developed in Java platform. The main features are the independency to the cell-level leakage model, and the circuit analysis using static logic values and signal probability.

3. CIRCUIT BEHAVIOR EVALUATION

The circuit evaluation is performed through a straightforward and recursive algorithm. The pseudo-code is shown in Fig. 2. The *cell_set* list holds the available cells for evaluation. The extraction of the logic behavior of each cell is then executed. After that, the signals are propagated throughout the entire circuit, from the primary inputs, according to the functionality of each logic gate. In order to determine the complete computing of the logic values in the circuit nodes, each cell has a counter, named *in_degree*, corresponding to its input number. When a value for one of its inputs is known, this counter is decremented. If it reaches '0', it means that all its input pins have been defined, and then such cell will be considered in the next iteration, being included in the *next_set* list. Otherwise, some input data is missing to define the output value and the cell is not evaluated yet.

```
Evaluate (list<cells> cell_set) {

 if cell_set is empty return;
 for each cell c in cell_set {
   evaluate c;

   for each cell n in netlists© {
     define_input(n, c.output);
     n.in_degree--;
     if n.in_degree == 0 add n to next_set;
     }
   }

 evaluate(next_set);
}
```

Figure 2 - Pseudo-code for circuit evaluation.

3.1. Logic evaluation

Given a cell expression and the logic values present at its input pins, the output value is obtained by using a *BDD* (Binary Decision Diagram)-based tool, developed by the team. This graph-like structure allows efficient logic evaluation.

3.2. Probability evaluation

In order to evaluate the probability of the logic value '1' in internal and output signals, three basic rules are followed: (1) the AND logic probability is obtained through the multiplication of the probabilities of all the signals involved; (2) the OR logic probability is obtained through the sum of the probabilities of all the signals involved; (3) the NOT logic probability is the complement of the probability of the input. Fig. 3 illustrates these rules. Their combination allows the probability estimation for all other gates, since the same input variable is not present in the Boolean equation more than once.

3.3. Leakage estimation

For each cell it is generated a transistor network according to the logic style chosen, e.g. conventional static CMOS, PTL and so on, since the leakage evaluation depends on the device arrangement. The circuit netlist together with the evaluated input vector allows both subthreshold and gate oxide leakage current predictions. The information of the minimum and maximum leakage values is stored, as well as the weighted average value. The weights applied correspond to the probability of each input vector to occur. Since the probability for each input signal is known, it is possible to determine the probabilities for each vector in the truth table of the cell. Making so, it is then possible to determine the minimum, maximum and average standby currents in the entire circuit.

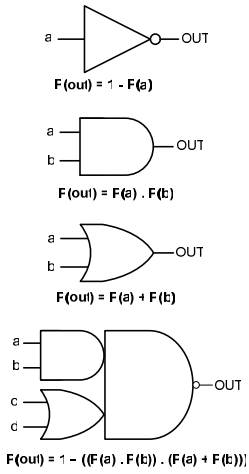


Figure 3 – Signal probability of logic gates.

4. SPICE CORRELATION

The simple C17 ISCAS benchmark circuit, depicted in Fig. 1a, is presented herein to illustrate the use of the LEE tool. The number at each signal indicates the iteration at which it was evaluated. An extraction of the report provided by the tool and containing the results is shown in Fig. 4

In order to validate the estimation method, the tool provides also the possibility to save individual cell netlist together with all possible input vectors. Thus, a Spice simulation is made easy and fast for correlation.

Hspice scripts are being developed in the sense to calculate individual cell leakage and the statistical average leakage of huge circuits, when all primary input combinations cannot be verified due to the number of signals.

```
<Inputs>
a       logic [1]       prob [0.5]
b       logic [1]       prob [0.5]
…
N       logic [0]       prob [0.5]
Total input #: N

<Outputs>
out1    logic [1]       prob [0.53125]
out2    logic [1]       prob [0.609375]
…
outM    logic [1]       prob [value]
Total output #: M

<Cell X1>
        Inputs: {i2=1, b=1}
        Output value: 0
        Output probability: 0.625
        Normalized leakage: 4.0
        Minimum leakage: 1.0
        Maximum leakage: 4.0
        Average leakage: 0.65625
…
<Cell Xi>
        Inputs: {a=1, c=0}
        Output value: 1
        Output probability: 0.75
        Normalized leakage: 2.5
        Minimum leakage: 1.0
        Maximum leakage: 4.0
        Average leakage: 0.5625

Total cell #: i

Lower bound circuit leakage: 6.0
Upper bound circuit leakage: 24.0
Average circuit leakage: 3.8
```

Figure 4 – Example of report provided by the tool.

5. CONCLUSIONS

This paper presented a tool which allows the leakage estimation at circuit level for steady state values, as well as according to the signal occurrence probability. It is suitable for different leakage prediction models which evaluate such a kind of consumption at cell level. In future works, the search of input vector that represents the minimum leakage dissipation in a circuit can be improved by this tool.

6. REFERENCES

[1] K. Roy, S. Mukhopadhyay and H. M.-Meimand, “Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits”, Proceedings of the IEEE, vol. 91, no. 2, Feb. 2003, pp. 302-327.

[2] R. X. Gu and M. I. Elmasry, “Power Distribution Analysis and Optimization of Deep Submicron CMOS Digital Circuit”, *IEEE JSSC*, vol.31, no.5, May 1996, pp.707-713.

[3] Z. Cheng, M. Johnson, L. Wei and K. Roy, “Estimation of Standby Leakage Power in CMOS Circuits Considering Accurate Modeling of Transistor Stacks”, *Proc. ISLPED*, Aug. 1998, pp. 239-244.

[4] S. Yang, W. Wolf, N. Vijaykrishnan, Y. Xie and W. Wang, “Accurate Stacking Effect Macro-modeling of Leakage Power in Sub-100nm Circuits”, *Proc. Int. Conf. on VLSI Design*, Jan. 2005, pp. 165-170.

[5] D. Lee, W. Kwong, D. Blaauw and D. Sylvester, “Analysis and Minimization Techniques for Total Leakage Considering Gate Oxide Leakage”, *Proc. DAC*, June 2003, pp.175-180.

[6] P. Butzen, R. Mancuso, L. Rosa Jr., A. Reis, R. Ribas, “Leakage Behavior in CMOS and PTL Logic Styles for Logic Synthesis Orientation”, Proc. IWLS, May 2007.