Leakage Power Estimation for Deep Submicron Circuits in an ASIC Design Environment

Rahul Kumar
Sasken Communication Technologies Ltd.
#21, Brigade Square, Cambridge Road,
Ulsoor, Bangalore-560008
rahulk@sasken.com

C.P. Ravikumar
Texas Instruments India Pvt Ltd
Wind Tunnel Road, Murgeshpalya
Bangalore-560017, India
ravikumar@ti.com

Abstract

Static power dissipation due to leakage current in transistors constitutes an increasing fraction of the total power in modern semiconductor technologies. Current technology trends indicate that the leakage contribution will increase rapidly. Developing power efficient products will require consideration of static power in early phases of design development. Design houses that use RTL synthesis based flow for designing ASICs require a quick and reasonably accurate estimate of static power dissipation. This is important for making early packaging decisions and planning the power grid. Keeping this in view, we propose a simple model which enables estimation of static power early in the design phase. Our model is based on the experimental data obtained from simulations at the design level: $\ln P_{leak}^{lib} = S^{lib} \ln \text{Cells} + C^{lib}$, where S^{lib} and C^{lib} are the technology-dependent slope and intercept parameters of the model and "Cells" is the number of cells in the design. The model is validated for a large benchmark circuit and the leakage power predicted by our model is within 2% of the actual leakage power predicted by a popular tool used in the industry.

Key Words: Leakage Power, Power Estimation, Linear Regression, Deep Submicron

1.0 Introduction

Rapid progress in the miniaturization of the VLSI technology requires that the supply voltage be scaled down to avoid hot-carrier effects in CMOS logic circuits. The circuit speed gets affected if the ratio of supply voltage V_{DD} to the threshold voltage V_{th} is less than 5 because the current driving capability of the gates decreases. In order to avoid degradation in speed, the threshold voltage is scaled down. This results in a sharp increase in the standby current. Thus, estimating the dynamic power of CMOS circuits as the only dominant power component is no longer valid. As has been predicted by Thompson et al [1], the static (or leakage) power will be of the same order of magnitude as the dynamic power in circuits of the future. In Figure 1, the trend of dynamic and static power for Intel's past few technology generations is shown; if the same trend continues, leakage power will be of the same order of magnitude as the dynamic power. The active power dissipation of the processor core varies significantly depending on the workload, whereas the leakage power dissipation is almost a constant. Since the system remains in the idle state the majority of the time, the contribution of leakage power is far more than that of dynamic power. Chandrakasan et al have made a similar observation after analyzing traces from X-servers [2]; the processor spends more than 95% of its time in standby mode.

Leakage power is thus becoming an important concern for VLSI circuits. Estimation of leakage power is required for gauging and optimizing the total power dissipation of the circuit. What is more, such estimators are most useful in early phases of the design life cycle. This reduces the design cycle time if any optimization at the architectural level is required for leakage power reduction. Most design houses now use an RTL synthesis based flow for designing application-specific integrated circuits. The requirement of such design houses is a simple estimator which is reasonably accurate and easy to incorporate in the ASIC design flow. This paper proposes such a simple estimator for leakage power.

The paper is organized as follows. We begin with a brief review of the leakage power estimation techniques in Section 2. In Section 3, a model for leakage power estimation is presented. In Section 4, the proposed model is validated for its accuracy with an example circuit. Finally, conclusions are presented in Section 5.

2.0 Related Work

The two main leakage mechanisms contributing to leakage power are: a) Subthreshold leakage: this component of leakage power has an exponential relationship with the threshold voltage [3]. b) *pn*-junction leakage: this component is a function of junction area and doping concentration and can be usually ignored [4].Therefore, subthreshold component of leakage current is dominant in submicron technologies and more focus has been paid to its estimation. For ultra deep submicron circuits gate oxide tunneling can be a major contributor to leakage power dissipation.

The work presented in the literature considers *logic states* at the inputs of the gates for leakage power estimation. The input pattern dependence of leakage

current for a 4-input NAND gate is illustrated in Table 1. In [5], Roy et al proposed a transistor-stacking model for leakage power estimation at the gate level. Their work uses BSIM2 model equations, [6] [7] for calculating the subthreshold current of the MOS transistors in the stack.

The stacking model calculates the leakage contribution of the transistors that are in the *off* state; the transistors that are *on* are assumed to be short circuits. For transistors that are connected in parallel, the following rules are applied:

- If both transistors are turned off, the leakage contribution of each transistor is calculated separately and added together.
- If one of the transistors is *on* and the other is *off*, then the leakage contribution of the *off* transistor is not taken into consideration.

The steps involved in calculating the subthreshold currents of the transistors that are turned off in a stack are given in [2]. After the subthreshold currents and the V_{DS} for each transistor have been estimated, the total leakage power is calculated using Eq. (1).

$$P_{leak} = \sum_i I_{DSqi}.V_{DSqi} \qquad (1)$$

I_{DSqi} is the quiescent leakage of the i^{th} transistor that is *off*, and V_{DSqi} is the drain-source voltage of the transistor. The leakage power for every gate is calculated in a similar manner. A similar approach is also presented in [8].

In [9], a genetic algorithm is used for estimating the minimum and maximum leakage power. The disadvantage of the approaches mentioned above is that they are not practical for large circuits. The methods of [5] and [8] rely on simulation; it is practically impossible to simulate a large circuit for all possible inputs in order to estimate leakage power. The circuits used in [9] are small in comparison to the present day VLSI circuits that have millions of transistors.

To address the issue of circuit size, we propose a simple model which predicts the leakage power based on the number of gates (or cells) in the design. We believe that most project managers will have a fairly good estimate of the number of gates in the design at an early stage in the project and can make use of our model to make an early estimation of the leakage power consumption of the chip.

3.0 Proposed Approach

The approach followed by us is based on experimentally estimating the leakage power for a set of small benchmark circuits and using statistical techniques for arriving at a simple linear regression model for leakage power.

3.1 Experimental Flow

In most design houses, it is common to use a synthesis-based ASIC design flow. We describe a flow that is easily practicable in such environments. See Figure 2. The circuit descriptions must be available in Verilog, VHDL, or EDIF formats. The timing constraints and operating conditions are fed to the synthesis tool through a script. The synthesis tool also takes as inputs a target standard cell library and, optionally, a library of pre-designed circuit blocks. In our own work, we used the Synopsys Design Compiler as the synthesis tool and the TSMC 0.18 μm technology standard cell library. The leakage power and the number of cells used by the design are extracted from the post-synthesis reports through *Perl* scripts. We used the ISCAS85 and ISCAS89 circuits in our work. The experiments were performed for three target libraries: slow, fast and typical. The three libraries are characterized for three different operating conditions (see Table 2).

The purpose of obtaining the leakage power for the three libraries is to find out which library exhibits maximum leakage power dissipation for a circuit. As an example, C17 benchmark circuit was synthesized for all the three libraries and the leakage power and the number of cells are shown in Table 3. The variation in the leakage power for the three libraries is dependent on the operating condition and the number of cells. The variation in the number of cells is due to the mapping algorithm used by the synthesis tool. The leakage power calculation implemented in Synopsys Power Compiler considers both subthreshold and *pn*-junction leakages for leakage power estimation.

The experimental data for the other benchmark circuits is shown in Table 4. The leakage power is experimentally obtained for the three target libraries for all the circuits. From Table 4, we observe that for all circuits, the leakage obtained for the *slow* library is the maximum.

3.2 Linear Regression Model

For each library, the *least square estimation technique* was used for arriving at a linear model of leakage power as a function of the number of cells. The linear model, described succinctly as $y = mx + c$, has two important parameters: the slope *m* and the intercept *c*. These parameters are data-dependent i.e. if the data points are scattered, then the percentage error in the prediction will be large. The slope and intercept, denoted here by S^{lib} and C^{lib} are shown in Table 5 for the three libraries. The leakage power model obtained is given by Eqn (2). Here, *Cells* is the number of cells in the design.

$$\ln P_{leak}^{\ lib} = S^{lib} \ln Cells + C^{lib} \qquad (2)$$

If we express the constant C^{lib} as $\ln \chi^{lib}$ then the above equation can be written as

$$P_{leak}^{lib} = \chi^{lib} Cells^{S^{lib}} \quad (3)$$

The model has two parameters, namely S^{lib} and C^{lib}, which can be estimated using the following equations.

$$S^{lib} = \frac{(\sum \ln P_{leak}^{lib} \ln Cells) - (\sum \ln P_{leak}^{lib} \sum \ln Cells)/n}{\sum (\ln P_{leak}^{lib})^2 - (\sum \ln P_{leak}^{lib})^2 / n}$$

$$C^{lib} = 1/n(\sum \ln Cells - \ln Cells . \sum P_{leak}^{lib})$$

Figures 3, 4, and 5 show the leakage power Vs the number of cells for all the libraries and their corresponding *linear models*. Interestingly, we see that the parameter S^{lib} is close to unity. In other words, the leakage power dissipation grows approximately *linearly* with the number of gates. The parameter χ^{lib} thus represents the approximate leakage power dissipation per cell. The percentage error in the predicted model for the three libraries is shown in Figure 6. The error was minimum for the slow library and maximum for the fast library. The plot in Figure 6 also shows the difference in the actual leakage power of the design and that predicted by the linear model.

4.0 Model Validation

The linear model presented in the previous section was validated for its accuracy. The example used for this purpose is a 24-bit DSP core obtained from Carnegie Mellon Univeristy [10]. The core is freely available for download in the public domain. The core is a derivative of the Motorola's DSP56002 DSP processor and implements a subset of instructions present in the Motorola's DSP56002 processor. The core implements Harvard architecture and can read two data memories in a single clock cycle. The core architecture has four units:

- Data Arithmetic and Logic Unit (ALU) is the heart of the CMU DSP core. The ALU contains the X, Y, A, and B registers along with the multiply-accumulate and adder units.
- Address Generation Unit (AGU) generates the addresses for accessing the data memories. One important aspect of the AGU is its ability to operate independently of the ALU.
- Program Control Unit (PCU) contains the program counter (PC) and the flag bits for controlling the operation of the CMU DSP.
- Bus Switch generates instructions to move data among the memories. It works in conjunction with the address bus to accomplish the data transfer.

The core was synthesized by targeting TSMC's 0.18µm standard cell library. The leakage powers obtained from the Synopsys Design Compiler for different units of the core and for the entire core are shown in Table 6. The leakage powers predicted by the linear model are also shown in Table 6. As can be seen from Table 6, the estimator errors on the positive side in most cases; for the case of the Bus Switch unit, the error is on the negative side i.e. the estimator was too optimistic. In fact, the largest percentage error in prediction is also for the Bus Switch unit. We also see that the leakage power estimates for smaller circuits are less accurate as compared to those for larger designs. Thus, the model can be practical for predicting leakage power well in advance for large designs.

5.0 Conclusions

Leakage power dissipation due to subthreshold leakage will become the most important component of overall power dissipation. The present trends in technology innovation, coupled with increasing number of transistors on chip, will increase the leakage power to magnitudes that are of the same order as dynamic power of the chip. While dynamic power decreases quadratically with the reduction in supply voltage, leakage power increases exponentially with the associated threshold voltage scaling. This creates a need for cost effective and fairly accurate models for estimating leakage power early in the design cycle of VLSI circuits. Such early predictions are necessary for planning the power during floorplanning and in making early decisions about the packaging of the chip. Estimation of leakage power is also useful in IDDQ testing of CMOS chips.

We presented a simple, practical model for leakage current estimation that can be useful to ASIC design houses. The advantage of the model is that it only requires information about the number of cells in the circuit. The technology and library dependent parameters, S^{lib} and C^{lib}, are embedded within the model and can be estimated using our technique. In order to use the proposed model at an early phase in the design life cycle, a project manager must input the expected number of gates in the design, which is relatively easier to provide. The model can also be extended to relate the leakage power to the area of the chip instead of the number of gates on the chip.

Acknowledgements :
This work was carried out when the authors were at IIT Delhi. We also thank Controlnet India Pvt Ltd. for the computational facilities to carry out this work.

References

[1] S. Thompson, P. Packan, and M. Bohr. **MOS Scaling: Transistor Challenges for the 21st Century**. In Intel Technology Journal, 3rd Quarter, 1998.

[2] A. Chandrakasan, I. Yang, C. Vieri, and D. Antoniadis. **Design Considerations and tools for Low-Voltage digital system design**. In Proceedings of the 33rd ACM/IEEE Design Automation Conference, 1996, pp. 728-733.

[3] C. Hu. **Device and Technology Impact on Low Power Electronics**. In Low Power Design Methodologies, Kluwer Academic, Boston, pp. 21-36, 1996.

[4] J. Rabaey. **Digital Integrated Circuits: A Design Perspective**. Addison-Wesley, Reading, MA, 1993.

[5] M.C. Johnson, K. Roy, and, D. Somashekhar. **A model for leakage control by transistor stacking**. Master's Thesis, Department of ECE, Purdue University, 1997.

[6] J. Sheu et al. **BSIM: Berkeley Short-Channel IGFET Model for MOS transistors**. IEEE Journal Solid-State Circuits, vol. SC-22, 1987.

[7] Avant! Corporation. **HSPICE User's Manual**. Vol. II, 1996.

[8] R.X. Gu, and M.I. Elmasry. **Power dissipation analysis and optimization of deep submicron circuits**. IEEE Journal of Solid-State Circuits, vol. 31, no. 5, May, 1996, pp. 707-713.

[9] Z. Chen, M. Johnson, L. Wei, and K. Roy. **Estimation of Standby leakage power in CMOS circuits**. In International Symposium on Low Power Electronics and Design, Monterrey, CA, August, 1998.

[10] Low Power Group. Electrical and Computer Engineering Department Carnegie Mellon University. http://www.ece.cmu.edu/lowpower/benchmarks.html

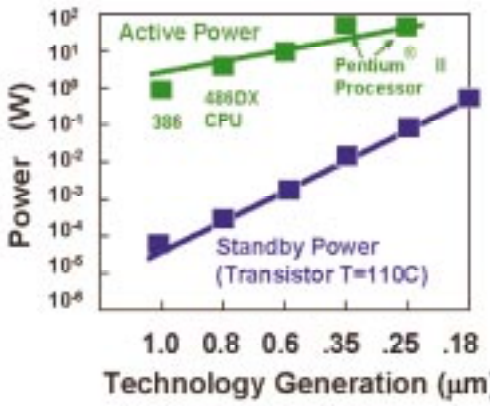

Figure 1: Dynamic and Leakage power for Intel's past few technologies

Table 1: Leakage current for 4-input NAND gate

Input Pattern	Leakage current (*nA*)
0111	9.96
1011	6.86
0001	0.98
0000	0.72
0101	0.0045
0101	0.0241
0011	1.71

Table 2: Operating conditions for 0.18um technology

Library Name	Voltage (V)	Temperature (^oC)
Slow	1.62	125
Fast	1.98	0
Typical	1.8	25

Table 3: Leakage power for C17 circuit, targeted for 0.18μm slow, fast and typical libraries

	Slow	Fast	Typical
Cells	OR2X2 (2) NAND2X1 (1) AND2X2 (3)	OR2X2 (1) NAND2X2 (1) INVX1 (1) AOI22X1 (1) AND2X2 (1)	OR2X2 (2) NAND2X2 (1) AND2X2 (3)
# **Cells**	6	5	6
P_{leak}**(nW)**	3.479	1.064	0.489

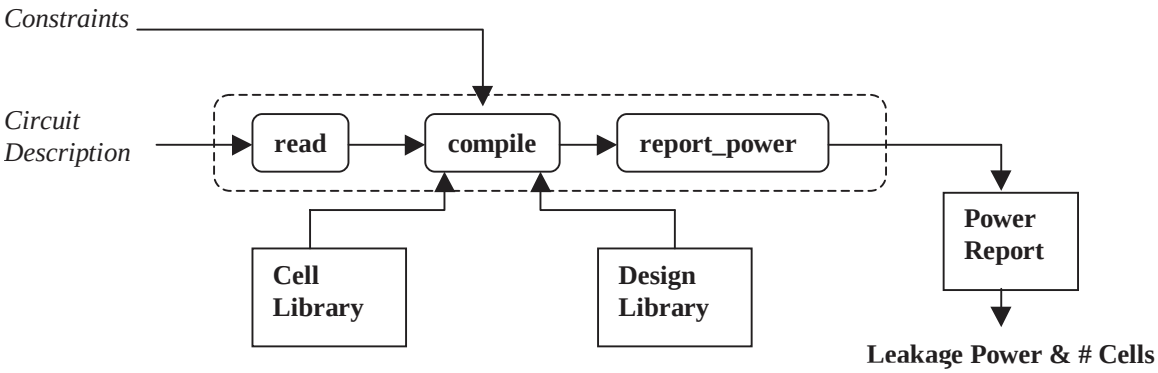

Figure 2: Leakage Power estimation flow

Table 4: Leakage Power and Number of Cells for the benchmark circuits

Circuit	P_{leak}^{slow} (nW)	$\#Cells^{slow}$	P_{leak}^{fast} (nW)	$\#Cells_{fast}$	$P_{fast}^{typical}$ (nW)	$\#Cells^{typical}$
c880	53.80	178	24.10	180	16.70	199
c5315	157.00	569	65.60	569	88.10	684
c1355	49.60	177	32.20	179	34.30	177
c1908	41.30	144	21.70	146	22.70	149
c2670	101.50	321	60.30	310	39.60	312
c3540	123.10	400	57.20	410	38.20	416
c6288	519.90	1849	216.90	1865	170.70	1839
s298	21.86	81	10.82	79	7.37	80
s344	21.36	86	10.85	85	7.97	86
s349	20.70	89	11.67	96	9.44	92
s386	25.02	81	9.97	85	4.92	85
s420	26.65	97	13.43	84	12.11	80
s382	29.87	114	15.59	113	10.85	112
s444	32.94	116	16.23	117	10.72	120
s400	32.92	119	17.16	117	11.46	119
s510	42.83	145	21.83	150	11.05	146
s526	36.69	130	16.62	130	11.90	129
s526n	34.54	128	16.07	129	12.61	129
s641	34.57	111	19.29	116	10.70	116
s713	34.57	111	19.29	116	10.70	116
s820	56.55	172	27.04	180	11.64	181
s832	60.25	179	26.01	187	11.00	185
s838	52.09	187	30.90	188	24.66	181
s953	81.02	254	36.66	272	22.68	281
s1238	97.08	310	34.44	316	21.38	318
s1196	97.19	312	39.78	306	21.38	311
s13207	491.41	1974	267.75	2000	288.96	2075
s1494	106.70	324	45.30	344	23.90	348
s5378	212.80	729	101.50	856	102.60	738
s15850	643.60	2090	347.00	2221	310.60	1961
s9234	150.80	571	79.60	669	82.20	586
s38417	1334.30	5302	746.10	5998	845.10	5208
s35932	2232.40	8381	961.20	5631	10.86.0	5701
s38584	2631.00	8836	1199.90	7587	793.10	6671

Table 5: Slope and Intercept for the three libraries

Library Name	Slope (S^{lib})	Intercept (C^{lib})
Slow	0.966	-1.032
Fast	0.946	-1.633
Typical	0.933	-2.212

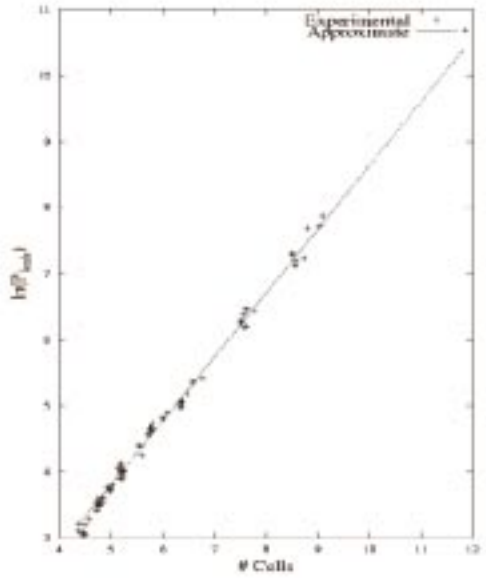

Figure 3: ln(P_{leak}) Vs ln(# Cells) for Slow library

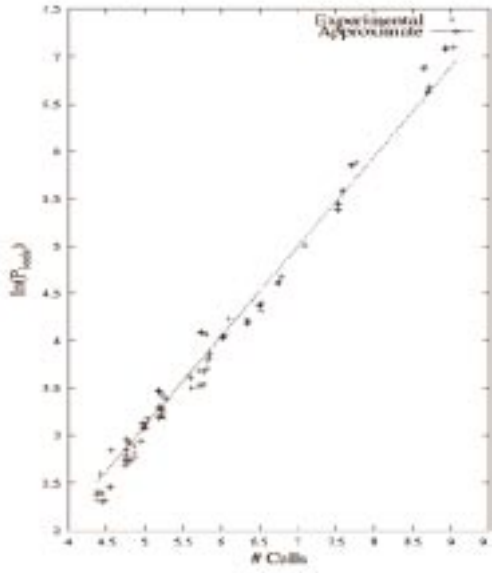

Figure 4: ln(P_{leak}) Vs ln(# Cells) for Fast library

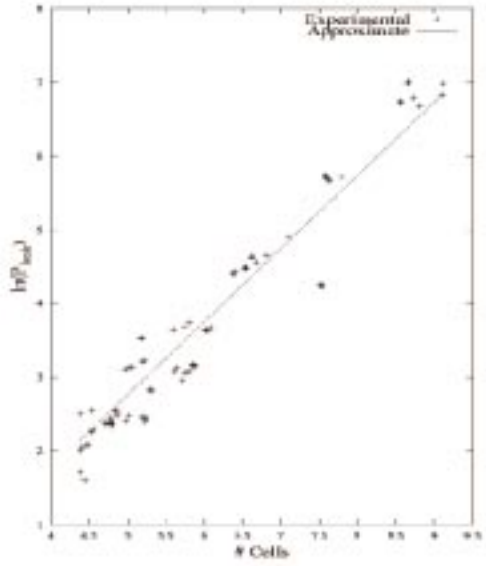

Figure 5: ln(P_{leak}) Vs ln(# Cells) for Typical library

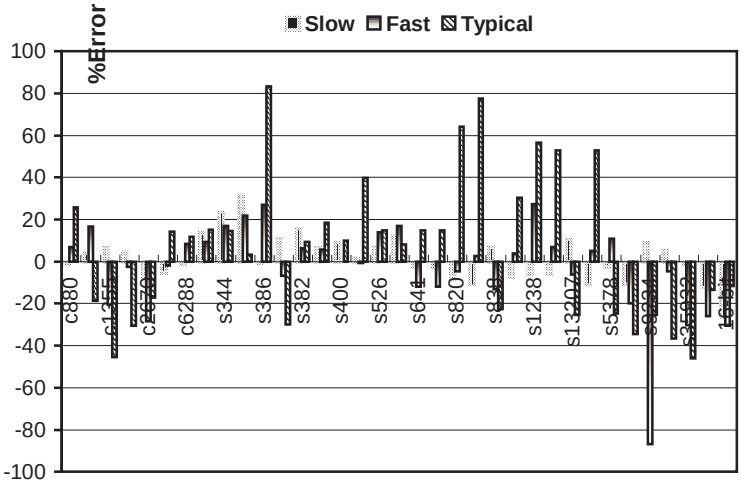

Figure 6: %Error of benchmark circuits for the three libraries

Table 6: Leakage Power for the CMU DSP Core: Comparison with the linear model

Component	# Cells	Leakage Power (µW)		% error
		Experimental	Linear Model	
ALU	7548	1.92	1.98	3.12
AGU	6500	1.59	1.71	8.22
PCU	3436	0.87	0.92	5.74
Bus Switch	458	0.16	0.14	-12.50
Core	17962	4.53	4.58	1.10