Runtime leakage power estimation technique for combinational circuits

Yu-Shiang Lin
EECS
University of Michigan
1301 Beal Ave. MI 48109
Tel: 734-709-6551
e-mail: yushiang@umich.edu

Dennis Sylvester
EECS
University of Michigan
1301 Beal Ave. MI 48109
Tel: 734-615-8783
e-mail:dennis@eecs.umich.edu

Abstract— This paper carefully examines subthreshold leakage during circuit operation (runtime) and develops a novel analysis technique to capture this important effect, which is currently ignored in traditional steady-state leakage calculation approaches. We implement novel dynamic and static estimation methods that provide significant speed improvements over full SPICE simulations and yield estimation errors of approximately 12% on average compared to more than 2X errors in steady-state based subthreshold leakage analysis.

I. Introduction

Rapid technology advancement in VLSI of recent years has resulted in high performance and high density designs. However, the increasing power dissipation and tight power budgets are now leading designer concerns. Though dynamic power has long been studied, its contribution to total power has diminished because the shrinking of physical dimension leads to the growth of leakage power[1].

Early works have shown that standby (or static) leakage power is strongly dependent on the input vectors[2, 3]. By modeling transistor stacks in CMOS circuits, subthreshold leakage can be computed using analytical expressions[4]. Analytical equations have been derived for simple transistor stacks and used to accurately measure the gate level leakage behavior. To reduce the complexity of transistor modeling, transistor states are solved by the Newton-Raphson method through a 3D lookup table[5]. The authors of that work also show that circuit-level estimation can be simplified by adopting the notion of dominant leakage states. As an alternative to the stack transistor approximation, effective transistor stacks were applied for different functional Unit Blocks (FUB) together with an effective width for total leakage power estimation[6]. Recently, the loading effect of CMOS logic circuits were investigated[7]. It was shown that fan-in and fan-out transistors can interact with the leakage component within a gate.

To obtain an early estimate of leakage power dissipation for RTL level synthesis, regression analysis based on gate count and width has been presented[8]. Another approach[9] shows a static estimation method that can calculate average leakage power efficiently.

Transient leakage behavior was discussed in [10] and it was shown that internal nodes of a transistor stack take on the order of μs to settle. Fig. 1 shows the subthreshold leakage of a NAND2 gate in a 90nm technology over time after the transition is completed. The first case is when the input of the top

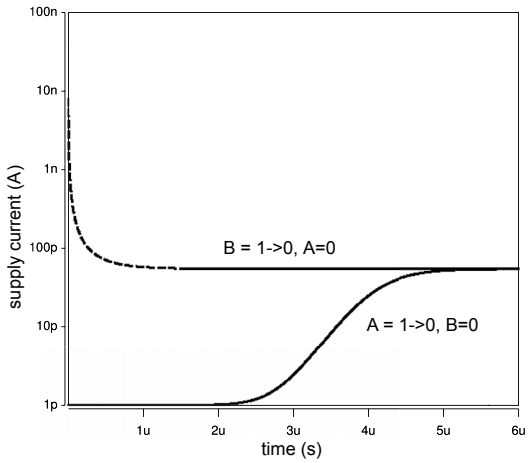


Fig. 1. Transient behavior of a NAND2 gate.

NMOS transistor A goes from Vdd to Gnd and the input of the bottom transistor B remains at Gnd. Since the top transistor turns off rapidly, the internal voltage is still higher than its steady state immediately after the transition. Thus, Vgs for the top transistor becomes negative and contributes far less leakage compared to the steady state. The opposite case happens when a remains unchanged and b undergoes a high to low transition. The coupled voltage will pull the internal node sufficiently below Gnd momentarily and cause orders of magnitude more current than I_{off} through the top transistor. Traditionally, this part of power loss is not included in either dynamic or leakage power estimation. It will be shown in this paper that the contribution of the *dynamic* leakage power cannot be neglected for modern combinational circuits.

The following sections are organized as follows. Section 2 demonstrates that runtime leakage cannot be well approximated by its steady state value. Gate-level characterization and circuit-level estimation will be covered in Section 3 for the proposed dynamic and static approaches to runtime leakage estimation. Section 4 will present and discuss simulation results of the approaches and the last section will draw conclusions.

II. Leakage as a function of time

Switching power is commonly estimated based on knowledge of switching capacitance [11] while short circuit current is analytically formulated [12, 13]. The impact of charging internal nodes has been considered in previous work [14], how-

1-4244-0630-7/07/$20.00 ©2007 IEEE.

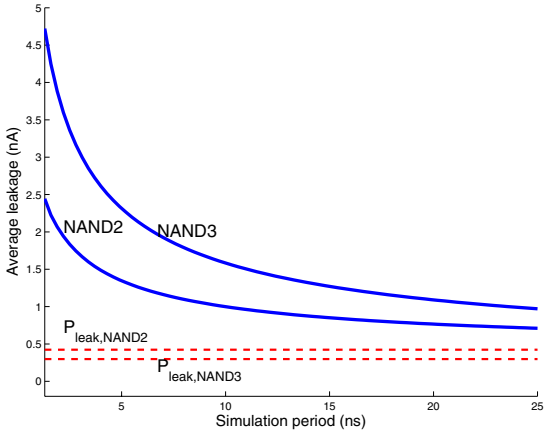


Fig. 2. Average leakage power with respect to simulation period with a fixed number of switching events.

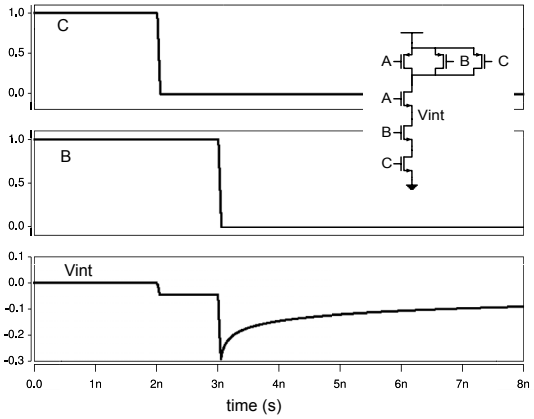


Fig. 3. Leakage power depending on history patterns for a NAND3 gate.

ever, it is very difficult to accurately compute all internal voltages. More importantly, an active circuit may not allow full transitions at its internal nodes before the next input switching event, such that this is difficult to classify as traditional dynamic power dissipation.

Due to the above considerations it is hard to cleanly separate the contributions of leakage power and other power dissipation components even using an industry standard library characterization tool such as Star-MTB[15]. In Star-MTB, both dynamic energy due to an input event and the leakage for that state are independent of time. Since switching power and short circuit power are consumed during the output switching event, lengthening the time elapsed between switching events should not impact their magnitudes. In this case, the difference in energy when running the same hardware with different frequency should solely come from the leakage. We will show that the difference is larger than steady state leakage and also decreases with time. In the following simulations, the reference period (period between input switchings) is set to 1ns and no explicit output loading is used to ensure fast transitions. 50 random input patterns are simulated for both NAND2 and NAND3 gates. Running at different period, the difference in total energy from reference period is computed and then divided by the difference in period to obtain the average (leakage) power of the additional period. Fig. 2 shows the average leakage power of the additional period converges to its steady state value when the period becomes large. This experiment shows that steady state analysis greatly underestimates the actual leakage result and also that NAND3 leakage is actually worse than NAND2 in practice, despite the fact that its steady state leakage is smaller. While more stacked transistors help in reducing steady state leakage, it also makes it more likely to produce higher leakage transient patterns. Further discussion of this effect is included in the next session when several characteristics of runtime leakage are discussed.

III. Proposed Dynamic Leakage Analysis Approach

As mentioned previously, the transient leakage component is not properly modeled in conventional library characterization flows. In order to more accurately estimate this component of power consumption, it is necessary to further investigate circuit parameters that have an impact on runtime leakage.

A. Gate-level characteristics of runtime leakage

Unlike its standby counterpart, runtime leakage depends not only on the current input vector S_0 but also on past input patterns $S_{-1}, S_{-2}, S_{-3} \ldots$. The reason is that when any internal node is floating, V_{GS} and thus the subthreshold leakage can not determined by the input vectors alone. The information from the previous state is helpful to identify the internal voltage. Fig. 3 demonstrates the condition when the top transistor of the stack is off and other transistors are on. When input C switches low, internal node Vint will be pulled below 0 due to capacitive coupling. If this is followed by another high-to-low transition by input B, Vint while already being floated previously will drop even lower because of another capacitive coupling event. This will momentarily increase the subthreshold leakage of the top transistor in order to recharge Vint. In this case, we need two past input patterns in addition to present state to precisely describe the behavior of such gate.

Runtime leakage is a function of time and input vectors as described in the above experiments, but is also impacted by input arrival times. Consider a case when both inputs of a NAND2 gate switch from high to low. If the top transistor input A switches slightly earlier than the bottom transistor input B, the resulting behavior is similar to the case where only B has transitioned while A remains at 1 (and vice versa). However, estimating the gate behavior becomes more complex when both inputs arrive nearly simultaneously. As shown in Fig. 4, the average leakage over the 5ns time period following a transition of input A at 1ns varies significantly when B leads A by $<$50ps. In this work, the subtle dependence on arrival time for multiple-input switching event will not be parameterized. All input transitions other than the latest one will be assumed to be happened at the previous cycle. This simplification can be justified by the fact that these scenario is relatively unlikely.

Glitches also contribute to power dissipation in a parasitic fashion. In addition to their previously studied impact on short-circuit and dynamic power consumption, glitches can change

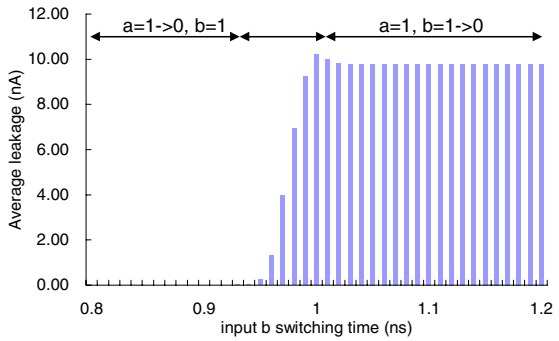


Fig. 4. Input arrival time dependent transient leakage for NAND2.

the internal node voltages as well as the *past state* of the gate. In a complex circuit with large logic depth or unbalanced logic paths, glitches can appear frequently and propagate inaccurate state information if it is not carefully tracked. Since we will not analytically calculate node voltage in this work, glitches will be determined by arrival time of the inputs. Ignoring glitch detection can cause an estimation error of approximately 5-20% in this work depending on circuit topology.

To characterize the runtime leakage behavior, it is difficult to find a simple yet accurate analytical expression that models most input patterns. However, it is helpful to simplify the problem to be solved and obtain an approximate analytical form for better understanding of the waveform. Consider two NMOS transistors in a stack where both are in the off state while the internal voltage connecting these transistors is lower than its steady state because of capacitive coupling (i.e., a high-to-low transition at one of the inputs has recently occurred which pulled down the internal voltage due to gate to source/drain overlap capacitance) . The top transistor will start to inject current into the internal node to pull up the voltage. At the beginning of this process, the current contribution of the bottom transistor can be neglected compared to the top transistor. Thus the following equation can be used to represent the charging process:

$$I_{sub} = \mu \cdot C_{OX} \cdot \frac{W}{L} \cdot exp[\frac{-V(t) - V_{th}}{nV_T}] = C_{int} \cdot \frac{dV(t)}{dt} \quad (1)$$

where C_{int} is the internal capacitance, V(t) is the internal voltage, $V_T = kt/q$ is the thermal voltage, W and L are transistor width and length, C_{OX} is the gate oxide per unit area, and μ is the electron mobility. The effects of drain-induced barrier lowering (DIBL) and therefore V_{DS} are neglected for simplicity. Eq. 1 can be further simplified as

$$\frac{dV(t)}{dt} = K \cdot exp[\frac{-V(t)}{nV_T}] \quad (2)$$

where K is given by $\mu \cdot (C_{OX}/C_{int}) \cdot (W/L) \cdot exp[-V_{th}/nV_T]$ and is treated as a constant here. Solving for this differential equation, it is found that $V(t)$ is a logarithmic function of time

$$V(t) = nV_T \cdot \ln[\frac{k}{nV_T} \cdot t] \quad (3)$$

which also implies that subthreshold leakage is proportional to t^{-1}.

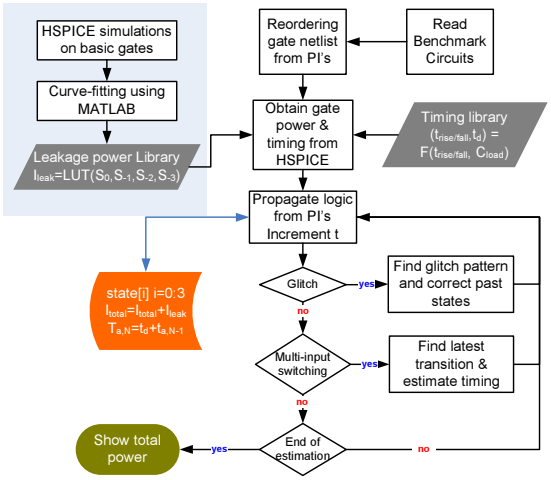


Fig. 5. Estimation flow for circuit level leakage estimation.

B. Dynamic leakage estimation

Fig. 5 demonstrates the flow we use to estimate subthreshold leakage during runtime.In a worst-case scenario, the number of patterns required to be characterized will grow exponentially with the number of inputs. For three input gates, there are a total of 4096 SPICE runs if four states are used. In this work, we consider three history states as a reasonable tradeoff between speed and accuracy. Using only two history states will introduce averagely another 10% of error to the final results.

Logic states for each node in the circuit are propagated from the primary inputs along with arrival times. If a multi-input switching event occurs (not at a primary input), only the latest transition in the clock cycle will be used to determine the current leakage pattern. Glitch events are also determined using the signal arrival times at the gate inputs. Since the algorithm goes through each gate exactly once for each clock period, the complexity of the dynamic estimation is $O(N_g \cdot N_t)$, where N_g is the number of gates in the circuit and N_t is the number of periods being simulated. To further reduce the computation time, we will develope a static estimation method to eliminate time dependence term so that the algorithm will be only linearly proportional to the circuit size.

C. Static estimation for leakage

Although dynamic estimation provides good accuracy by tracking cycle-by-cycle behavior of signals in a circuit, a static estimation technique may be preferable in many cases. In high level synthesis, an input independent analysis for leakage power is helpful to reduce runtime. Correlation between logic signals will be ignored in our work for simplicity.

From the previous discussion, runtime leakage is highly dependent on the logic state and activity rate of each input. Thus, the state probability and transition probability of each node has to be computed in the first step. Input state at time t is denoted as x_i^t where the superscript t will be ignored when only the present states are considered. Consider the case that the input state probability $P(x_i = 1)$ and the input probability of switching states $P(x_i^t, x_i^{t+1})$ are given for input i. The output

state probability of being 1 can be written as

$$P(x_o = 1) = \sum_{o_i \in min(o)} P(x = o_i) \quad (4)$$

where $x = x_1 x_2 x_3 \ldots$, x_o is the logic output of the gate, and $min(o)$ is the set of minterm expressions of the output logic. It is noted that due to the input independent assumption, the state probability of o_i can be written as

$$P(o_i) = P(x_1, x_2, \ldots x_n) = P(x_1) \cdot P(x_2) \ldots P(x_n) \quad (5)$$

where n is the total number of inputs. The switching probability of the output can be expressed using the conditional probability of previous state

$$P(x_o^t, x_o^{t-1} : x_o^t \neq x_o^{t-1}) = \sum_{o_i \in min(o)} P(x^{t-1} = o_i) \cdot P(x_o^t \neq x_o^{t-1} | x^{t-1}) \quad (6)$$

where x_o^t is the logic output at time t. Again, to simplify the above equation we assume independence of input switching probabilities. Then $Pr(x_o^t \neq x_o^{t-1} | x_o^{t-1})$ can be expressed as

$$P(x_o^t \neq x_o^{t-1} | x^{t-1}) = \prod_i P(x_o^t \neq x_o^{t-1} | x_i^{t-1}) \quad (7)$$

$Pr(x_o^t \neq x_o^{t-1} | x_i^{t-1})$ can be easily determined given the gate function and switching probability of input i. Finally, total leakage power can be estimated as

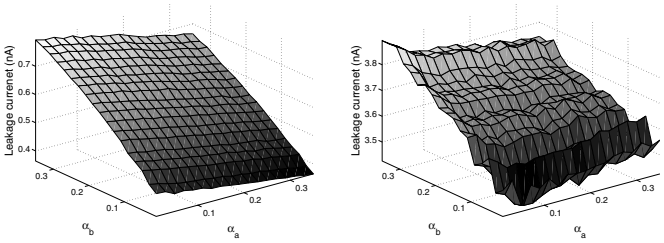
$$\sum_t \sum_{g \in G} Leak(P(x^t, x^{t-1})) \quad (8)$$

G is the set of gates in the circuit, and $Leak(P(x^t, x^{t-1}))$ is the input switching probability rate dependent leakage power. $Leak(P(x^t, x^{t-1}))$ can be computed using the previously discussed dynamic approach. 100000 random input patterns are applied to generate each data point of $Leak(P(x^t, x^{t-1}))$ lookup table for more equal probability patterns.

Fig. 6 shows the leakage power with respect to the activity rate of both inputs (α_a for the top transistor and α_b for the bottom transistor) of a NAND2 gate, in the case of a medium-Vt implementation and a low-Vt implementation. The leakage rises uniformly with α_b while increasing α_a slightly decreases leakage. These trends can be explained by two different leakage behaviors originally shown in Fig. 1. Comparing the different Vt devices, since low Vt devices have larger steady state leakage they will also stabilize their internal voltages much faster compared to medium Vt devices. Noting the z-axis values in fig. 6(b), there is only a 15% deviation from the steady state leakage estimation for low Vt devices. Therefore, for dual-Vt designs that are generally dominated by medium Vt devices [16], the runtime leakage contributed by low Vt devices can be neglected with limited error.

IV. Simulation setup and results

To validate the results of our proposed estimation methods, HSPICE simulations are used as the golden reference. With many potential glitches in a complex circuit, it is difficult to accurately separate runtime leakage power from other power dissipation sources based on a straightforward current measurement. We therefore define a current measurement method that can be realized in HSPICE simulation and reflects the nature of runtime leakage power.



(a) (b)

Fig. 6. Average leakage current as a function of input activities for a NAND2 gate using (a) medium Vt devices (b) low Vt devices.

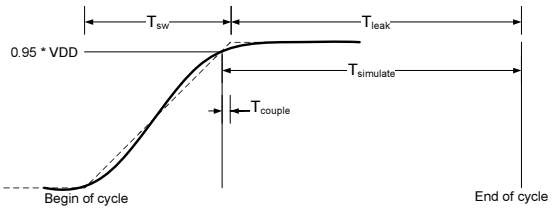


Fig. 7. Demonstration of current measurement setup in HSPICE simulation.

A. Current measurement

Fig. 7 shows the basics of the current measurement scheme. First, leakage current is measured as the current through the first off-transistor in the stack relative to the output node. In this way, switching power consumption will be excluded from the measurement result. For each cycle of simulation, leakage current is measured only after the last input transition. The measurement threshold is taken to be 95% of the full swing to avoid measuring short-circuit current (note that taking 100% is not feasible since signals transition their final 5% very slowly). However, power consumption arising from coupling between gate and source/drain due to the remaining input signal transition will introduce unwanted current to the leakage measurement. Thus, we pre-characterize this coupling-driven current using an ideal transition (shown by the dotted line in Fig. 7) so that it can be calibrated out from the results. Although this approach is an indirect measurement, it separates runtime leakage current from the others with a reasonable approximation. It is noted that in this setup, the leakage from the beginning of the cycle to the end of last transition is not included. Since the period is relatively small compared to other idling clock cycles and the dynamic leakage is generally much lower in this period, the contribution to the total leakage can be neglected.

B. Simulation setup

Table I summarizes the simulation setup for this work. Due to the larger subthreshold leakage current in low Vt devices, the difference between steady state and runtime current is a lot smaller than in a nominal or high Vt device. Therefore high Vt devices are used to examine the accuracy of our leakage estimation methods at room temperature (temperature effects will be discussed later). A primary input activity rate of 12.5% is used, meaning that one rising and one falling event occurs every eight clock cycles which is typical of VLSI circuits[17].

TABLE II
SUMMARY OF SIMULATION RESULTS. (* SIMULATION RESULTS OF 250 INPUT PATTERNS)

	Clock period(ps)	HSPICE I_{avg}(nA)	Dynamic method I_{avg}(nA)	Error compared to HSPICE(%)	Static method I_{avg}(nA)	Error compared to HSPICE(%)	Steady state I_{avg}(nA)	Error compared to HSPICE(X)
c17	1000	8.52	7.57	-11.16	8.12	-4.66	3.51	2.43
c432	1000	236.97	210.69	-11.09	255.83	7.96	80.41	2.95
c499	1000	607.14	712.14	17.29	399.90	-34.13	230.48	2.63
c880	1000	604.54	583.74	-3.44	675.75	11.78	184.17	3.28
c1355	1000	527.10	641.82	21.76	595.92	13.06	258.16	2.04
c1908	1500	461.39	424.98	-7.89	440.46	-4.54	202.72	2.28
c2670	1500	930.5	721.2	-22.49	815.74	-12.33	426.68	2.18
c3540	1500	970.87	1075.62	10.79	1179.59	21.50	450.72	2.15
c5315*	1500	1407.61	1418.52	0.78	1466.14	4.16	809.65	1.53
c6288*	2500	2058.25	1894.82	-7.94	1938.39	-5.82	1177.33	1.75
c7552*	1500	1520.35	1650.13	8.54	1796.31	18.15	992.75	1.74
$\|Avg.\|$				11.2		12.6		

TABLE I
SUMMARY OF SIMULATION SETUP

Technology	Industrial 90nm CMOS
Cell libraries	INV, NAND2, NAND3 NOR2, NOR3
Benchmark circuits	ICSAS '85
PI rise/fall time	50ps (0-100%)
Input pattern	Randomly generated with activity rate of 12.5%
Number of patterns	500 or 250

TABLE III
RUN TIME COMPARISON IN SECOND

	HSPICE	dynamic method	static method
c17	691.23	0.88	0.17
c432	18793.67	16.6	0.16
c499	87338.57	49.28	0.18
c880	68730.12	40.54	0.17
c1355	109726.36	57.68	0.19
c1908	87812.18	45.45	0.24
c2670	434945.44	95.20	0.22
c3540	442573.41	103.49	0.24
c5315	553214.50	92.54	0.29
c6288	1452735.07	149.81	0.38
c7552	671991.82	211.85	0.33

500 random input samples are used in the HSPICE simulations except for the larger benchmarks for which 250 cycles are simulated; large runtimes prevent running more than this.

C. Simulation results

Table II shows SPICE simulation results and a comparison with our dynamic and static estimations. On average, estimation error is 11.2% and 12.6% over the benchmark circuits for the dynamic and static estimators, respectively. In Table III, the computation time using the dynamic estimation technique is shown to grow linearly with circuit size and handles circuit of 10K gates reasonably well. The static estimator, which is based on the dynamic approach results, can significantly improve the runtime speed at the expense of time to characterize the activity rate dependent leakage for each gate. We also show that under reasonable switching frequency (125MHz for a clock period of 1ns), steady state leakage is >2X different than the runtime analysis. Computation time is based on Intel Pentium 4 3.4GHz platform.

The sources of error in our approaches are mostly contributed by signal glitching, multiple input switching, and small input variations. Our leakage library by design considers the input pattern history on a cycle by cycle basis, which means that only one state per cycle is assumed. So although the dynamic estimator can detect the correct past states caused by a glitch, the relative timing between transitions will not match those characterized in the leakage library. Similar to a multiple input switching event, there is a region during which the leakage is strongly sensitive to the glitch amplitude. The last reason is due to the simulation setup. Small input disturbance within 5% of the power rails is neglected by definition as stated in Section A. Thus, when a small input disturbance and a regular input transition happen at nearly the same time, the coupling current will not be calibrated out from the HSPICE simulation results. This is a rare scenario and should not strongly impact accuracy. Overall, it was found that glitch modeling is still the dominant component of estimation error.

It is instructive to show average runtime leakage as a function of switching frequency. Fig. 8 shows the trend of increasing leakage current as switching frequency rises for the simple ISCAS c17 circuit with 10000 input patterns. The figure also shows that the leakage estimated using our approach converges to steady state when there is very low activity, as expected. In some cases runtime leakage is slightly lower than steady state leakage. This can be explained by a disproportionate number of patterns being exercised that serve to lower the subthreshold leakage from its steady state (such as the situation when B is low and A switches low shown in Fig. 1).

Although we did not mention temperature as a factor impacting runtime leakage in previous sections, its impact on leakage in general is well known to be significant. Fig. 9 plots the

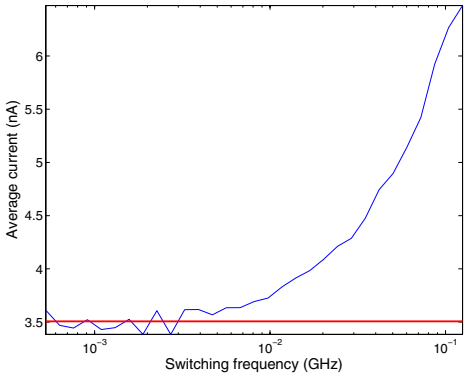


Fig. 8. Switching frequency vs. leakage current.

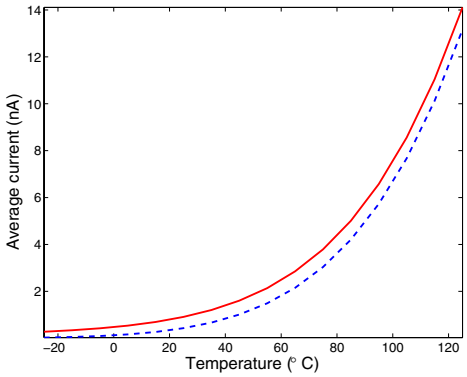


Fig. 9. Leakage current as a function of temperature. (Solid line: runtime leakage, dotted line: steady state leakage). Note the fixed difference between the two curves throughout the temperature range.

dependence of average leakage on temperature for a NAND2 gate. Runtime leakage (solid line) closely follows the curve of steady state leakage (dotted line) over a wide range of temperature. When temperature increases subthreshold current rises exponentially, implying that runtime current will be much larger but also achieve steady state faster. This results in similar power consumption as at the lower temperature. This same reasoning was observed in the low Vt and medium Vt discussion in a previous section. As a result, we can analyze runtime leakage at a single temperature point and then employ a temperature-dependent steady state leakage to estimate runtime leakage at different temperatures without re-characterizing it.

V. Conclusions

In this work, transient behavior of leakage due to transistor stacks is discussed. Due to a number of effects such as capacitive coupling from the input to internal nodes, the actual runtime leakage of a gate may be either much more or less than the steady state leakage estimation depending on the pattern history. We developed a dynamic estimation method that tracks recent pattern history and obtains leakage using table lookup from a pre-characterized library. To further reduce computation time and enable higher level analysis, we also propose a static method that uses expected activity rates to find the corresponding leakage of a gate. Both methods show good accuracy and significant speedups compared to HSPICE simulations.

Acknowledgements

This work was supported by Semiconductor Technology Academic Research Center and MediaTek International Student Fellowship.

References

[1] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, pp. 23–29, Jul 1999.

[2] J. P. Halter and F. N. Najm, "A gate-level leakage power reduction method for ultra-low-power cmos circuits," *Custom Integrated Circuit Conf.*, pp. 475–478, May 1997.

[3] Z. Chen, M. Johnson, L. Wei, and K. Roy, "Estimation of standby leakage power in cmos circuits considering accurate modeling of transistor stacks," *ISLPED*, pp. 239–244, Aug 1998.

[4] R. X. Gu and M. I. Elmasry, "Power dissipation analysis and optimization of deep submicron CMOS digital circuits," *Journal of Solid State*, vol. 31, pp. 707–713, May 1996.

[5] S. Sirichotiyakul, T. Edwards, C. Oh, R. Panda, and D. Blaauw, "Duet: An accurate leakage estimation and optimization tool for dual-vt circuits," *Trans. on VLSI*, vol. 10, pp. 79–90, Apr 2002.

[6] W. Jiang, V. Tiwari, E. de la Iglesia, and A. Sinha, "Topological analysis for leakage prediction of digital circuits," *ASPDAC*, pp. 39–44, Jan 2002.

[7] S. Mukhopadhyay, S. Bhunia, and K. Roy, "Modeling and analysis of loading effect in leakage of nano-scaled bulk-cmos logic circuits," *DATE*, pp. 224–229, Mar 2005.

[8] R. Kumar and C. Ravikumar, "Leakage power estimation for deep submicron circuits in an asic design environment," *ASP-DAC*, pp. 45–50, Jan 2002.

[9] E. Acar, A. Devgan, R. Rao, Y. Liu, H. Su, and J. B. Sani Nassif, "Leakage and leakage sensitivity computation for combinational circuits," *ISLPED*, pp. 96–99, Aug 2003.

[10] M. C. Johnson, D. Somasekhar, and K. Roy, "Models and algorithms for bounds on leakage in CMOS circuits," *Trans. on CAD*, vol. 18, pp. 714–725, June 1999.

[11] F. N. Najm, "A survey of power estimation techniques in VLSI circuits," *Trans. on VLSI*, vol. 2, pp. 446–455, Dec 1994.

[12] S. R. Vemuru and N. Scheinberg, "Short-circuit power dissipation estimation for CMOS logic gates," *Trans. on Circuit and System*, pp. 762–765, Nov 1994.

[13] S. Deng and A. J. Al-Khalili, "A technology portable analytical model for dsm cmos inverter short-circuit power estimation," *NEWCAS*, pp. 101–104, June 2004.

[14] A. Bogliolo, L. Benini, and B. Ricco, "Power estimation of cell-based CMOS circuits," *DAC*.

[15] Synopsys, *Star-MTB: The most automated and accurate cell characterization and model generation tool on the market*.

[16] T. Karnik, Y. Ye, J. Tschanz, L. Wei, S. Burns, V. Govindarajulu, V. De, and S. Borkar, "Total power optimization by simultaneous dual-Vt allocation and device sizing in high performance microprocessors," *DAC*, pp. 486–491, June 2002.

[17] G. Gerosa, M. Alexander, J. Alvarez, C. Croxton, M. DAddeo, A. R. Kennedy, C. Nicoletta, J. P. Nissen, R. Philip, P. Reed, H. Sanchez, S. A. Taylor, and B. Burgess, "A 250-MHz 5-W PowerPC microprocessor with on-chip L2 cache controller," *Journal of Solid State*, pp. 1635–1649, Nov 1997.