A Gate-Level Leakage Power Reduction Method for Ultra-Low-Power CMOS Circuits†

Jonathan P. Halter and Farid N. Najm

ECE Dept. and Coordinated Science Lab.
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

In order to reduce the power dissipation of CMOS products, semiconductor manufacturers are reducing the power supply voltage. This requires that the transistor threshold voltages be reduced as well to maintain adequate performance and noise margins. However, this increases the subthreshold leakage current of p and n MOSFETs, which starts to offset the power savings obtained from power supply reduction. This problem will worsen in future generations of technology, as threshold voltages are reduced further. In order to overcome this, we propose a design technique that can be used during logic design in order to reduce the leakage current and power. We target designs where parts of the circuit are put in "standby" mode when not in use, which is becoming a common approach for low power design. The proposed design changes consist of minimal overhead circuitry that puts the circuit into a "low leakage standby state," whenever it goes into standby, and allows it to return to its original state when it is reactivated. We give an efficient algorithm for computing a good low leakage power state. We demonstrate this method on the ISCAS-89 benchmark suite and show leakage power reductions of up to 54% for some circuits.

1. Introduction

As VLSI devices have grown in complexity and density, their power consumption has become a major design concern. High power consumption exacerbates reliability problems by raising the device operating temperature. It also increases the current density in the supply lines, causing greater electromigration problems. High power consumption also impacts battery-powered portable devices by requiring either large battery packs or unacceptably short operating time. These issues have forced designers to aggressively pursue low-power design methodologies.

It is well known that the power dissipation is directly proportional to the square of the power supply voltage, while it is proportional only to the first power of the capacitance and frequency ($P_{av} = CV^2f$). Thus, much work has been done on the technology side to reduce the power supply voltage from 5 V to 3.3 V and below. This trend will likely continue in the future, and we may soon see 1 V or lower supplies [1]. However, reduction in the supply voltage has a negative effect on circuit performance. Propagation delays through logic gates increase as the supply voltage decreases, and the overall noise immunity of the circuit decreases.

† This work was supported by Intel Corp. and by a National Science Foundation Graduate Fellowship.

To reduce these undesirable effects, threshold voltages have also been lowered [1]. However, lower threshold voltages increase the leakage power dissipation caused by subthreshold conduction in the MOSFETs. For circuits with very low threshold devices, the standby (leakage) power is no longer negligible [1] and can even dominate the active and short-circuit power. Future circuits promise lower threshold voltages and even greater standby power dissipation. Thus, methods to manage or reduce the leakage power are needed.

A number of leakage reduction methods have already been proposed. Methods by Horiguchi, et al., [3] and Mutoh, et al., [4] use circuit and process level changes to reduce the leakage power. In this paper, we propose a novel design method that can be used during *logic design* in order to reduce the leakage power of CMOS circuits. This approach is useful for designs in which parts of the circuit are put into idle or sleep modes by holding the clock fixed at either low or high. Such clock-gating schemes are quite common in low-power designs [2], so the proposed approach should be widely applicable.

Our approach is based on the observation that a CMOS logic gate dissipates leakage current in steady state which is dependent on the gate input state. Thus, given a multi-gate logic circuit, we modify the original logic design, using minimal additional circuitry, to force the combinational logic into a low-leakage state during an idle period. To find such a low-leakage state, we have developed an efficient algorithm that determines a good (low-leakage) input vector using a sampling of random vectors. The size of the sample set is determined a priori using user-supplied quality measures. The algorithm is based on a gate library which we have characterized for leakage current. We have demonstrated this method on the ISCAS-89 benchmark circuits and shown leakage power reductions of up to 54%.

Our design methodology is given in section 2. Section 3 describes the gate library characterization process. An input vector selection technique is developed in section 4. Suitable latch designs are shown in section 5, and section 6 shows the results of our method.

2. Design Methodology

We propose to use a natural characteristic of static CMOS gates in order to reduce the leakage power. This is based on the observation that individual CMOS gates show a variation in the leakage power based on the input vector, as seen in Table 1. Larger combinational circuits composed of many static CMOS gates exhibit similar variation, as shown in Fig. 1, which displays the leakage power histogram for a 119-gate circuit over a population of 100,000 randomly chosen input vectors. The leakage power for the circuit varies by a factor of two from the minimum leakage power to the maximum. Thus the leakage power is dependent on the input vector applied to the circuit.

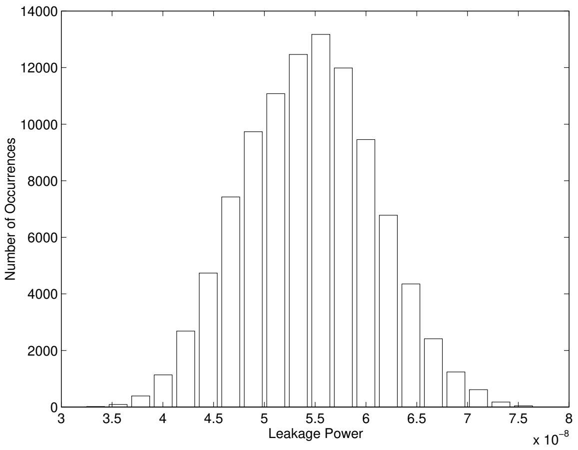

Figure 1. S298 leakage power histogram.

Modern low-power designs make extensive use of clock-gating. Clock-gating is a logic design method in which the clock is disabled to parts of the circuit during periods when they are not required to execute. These parts are said to be in standby mode, also called sleep mode or idle mode. The power supplies to these parts are *not* turned off, because of the performance and noise penalties that would result if this were done. Thus, whenever a circuit is put in standby, the latches or flip-flops inside it maintain the last state they were in. As a result, the circuit dissipates leakage power during standby corresponding directly to the logic *state* in which it was left.

Using minimal additional circuitry, we propose to modify the logic design so that whenever a circuit is put in standby, its internal state is set to a low-leakage state, preferably the lowest-leakage state possible. When that circuit is reactivated, the circuit will be returned to its last valid state. If the idle periods are long enough, this should lead to significant power reductions.

In section 4, we will present an efficient algorithm for finding a good low-leakage state. This has to be done only once, during the logic design phase. Once a good state vector has been determined, it can be multiplexed onto the logic inputs. Upon entering sleep mode, the state bits are selectively forced to 0 or 1, depending on the latch design used. Suitable latch designs in two common design styles are shown in section 5.

The search algorithm for a low-leakage input vector, to be given in section 4, depends on being able to estimate the leakage for a candidate vector. A circuit-level simulator, such as SPICE, can be used to do this, but a more efficient solution is possible since the leakage depends only on the steady state node values. Thus, it is more efficient to build library models for logic gates that give the leakage current drawn by a gate for each of its input combinations. We have done this by building look-up table models as in Table 1 for several kinds of logic gates and then running a simple steady state logic simulation in order to determine the leakage for a given circuit input vector.

3. Gate Library Characterization

In order to efficiently estimate the leakage of a large combinational circuit for a given input pattern, we need to develop models for logic gates that give the leakage current drawn by the gate for each of its input patterns. Using a circuit-level simulator, such as SPICE, and an appropriate MOSFET device model, we determine the leakage current for each input pattern and log the results into a look-up table. As an example, Table 1 shows the look-up table for a 2-input NAND gate. The values in this table are based on data which was obtained from a major semiconductor manufacturer, which we will refer to as "company A," and is derived from a 0.5 μm CMOS process.

Table 1.
Leakage current for a 2-input NAND gate.

Input Vector	Leakage Current
0 0	3.944×10^{-14} A
0 1	1.525×10^{-13} A
1 0	1.365×10^{-13} A
1 1	4.568×10^{-14} A

4. Input Vector Determination Technique

Consider a combinational circuit whose input nodes are state bits of an overall sequential circuit which will be put in standby mode. We need to choose an input vector for the combinational circuit that causes it to dissipate very low leakage power. The "search problem" for the vector that gives the *least* leakage power is a very difficult one because of the potentially huge size of the search space. Furthermore, it is not absolutely necessary to find this minimizing vector. Instead, one is interested in a vector that gives a significantly lower value of leakage and which can be found efficiently. We have developed an algorithm to find such a vector based on a process of random sampling. Randomly chosen vectors are applied to the circuit and the leakage due to each is monitored, and the vector which gives the least observed leakage value is reported. It will be seen that this relatively simple approach works well, in the sense that a relatively small number of vectors is enough to come close to the lowest leakage current that would be observed from a much larger number of vectors.

Clearly, the number of vectors to be applied determines the "quality" of the resulting solution. In the following, we will derive a simple result (9) which gives the number n of vectors required, a priori, for a given desired "quality" of the solution. To make this terminology more specific, we need to invoke a probabilistic view of the space of the Boolean vectors, as follows.

Consider the set of all possible input vectors to the circuit, and consider an experiment by which a vector is chosen at random, with all vectors having the same probability of being chosen. Thus, the Boolean space becomes a probability *sample space*. If v is an input vector, define a random variable (RV) $X(v)$ to be the leakage current drawn by the circuit when v is applied at its input under steady state. Let $f(x)$ and $F(x)$ be the probability density function (pdf) and the cumulative density function (cdf) of X, respectively. A typical $f(x)$ will be similar to the leakage power histogram in Fig. 1.

If we choose n input vectors $v_1, v_2, \ldots, v_n$ at random, by making an independent choice every time, the leakage current obtained in each trial, is a sample of an independent RV $X_i = X(v_i)$, $i = 1, \ldots, n$, with pdf $f(x)$. Since they are independent, the set $\{X_1, X_2, \ldots, X_n\}$ constitutes a *random sample*.

Define a new RV $Y = \min(X_1, X_2, \ldots, X_n)$, so that Y is the (random) lowest leakage value over n trials. The RV Y is called the first *order statistic* of the random sample. Suppose we are able to establish that, for sufficiently large n, we have:

$$\mathcal{P}\{F(Y) \leq \epsilon\} \geq \alpha \tag{1}$$

where $\mathcal{P}\{\cdot\}$ denotes probability, $0 \leq \epsilon \leq 1$ with $\epsilon \approx 0$, $0 \leq \alpha \leq 1$ with $\alpha \approx 1$, and $F(\cdot)$ is the cdf of X as stated above. This *probability statement* (1) would allow us to make a *statistical confidence* statement about an *observed value* (or *sample*) of Y, which we will denote by y, as follows: *with better than α confidence, we have $F(y) \leq \epsilon$.* This terminology is commonly used in statistics. Using the definition of the cumulative density function, i.e., $F(y) = \mathcal{P}\{X \leq y\}$, this would lead to the statement that *with better than α confidence, we have*:

$$\mathcal{P}\{X \leq y\} \leq \epsilon \tag{2}$$

Thus, if we can determine a value of n for which (1) holds, then an observed value y of the RV Y will have the following special property: *with better than α confidence, y is a leakage current value such that only a very small fraction (less than ϵ) of vectors in the Boolean space have leakage currents less than y.* If we keep track of which input vector generated the leakage value y, that vector would be a good candidate for the desired low-leakage input vector. The *quality* of this vector would be determined by α and ϵ.

We now determine the value of n for which (1) holds for a given ϵ and α. This requires the distribution of the RV $Z = F(Y)$, which is known to have a *beta distribution* from the field of reliability analysis and can be obtained from the rank distribution [5]:

$$g(z)dz = \frac{n!}{(j-1)!(n-j)!} z^{j-1}(1-z)^{n-j} dz \tag{3}$$

with $j = 1$ and $0 \leq z \leq 1$. Integrating this pdf to determine the cdf of Z yields:

$$\begin{aligned} \mathcal{P}\{F(Y) \leq \epsilon\} &= \int_0^{\epsilon} \frac{n!}{0!(n-1)!} z^0 (1-z)^{n-1} dz \\ &= 1 - (1-\epsilon)^n \end{aligned} \tag{4}$$

Setting this to be greater than α, according to (1), leads to the desired result:

$$n \geq \frac{\ln(1-\alpha)}{\ln(1-\epsilon)} \tag{5}$$

Thus, in order to have 95% confidence ($\alpha = 0.95$) that less than 5% ($\epsilon = 0.05$) of the vector population has leakage current which is less than the least observed leakage (y) from n trials, we need $n \geq \lceil \ln(0.05)/\ln(0.95) \rceil = 59$. For $\alpha = 99\%$ confidence and $\epsilon = 1\%$ error tolerance, we need at least 458 trials. In general, given a desired confidence α and tolerance ϵ, one can determine n a priori using (5).

5. Latch Designs

The circuitry used to force the low-power input vector onto the combinational logic should add as little speed and area overhead as possible to the design. Solutions such as pass-gate multiplexers and CMOS NAND and NOR gates can be used to force the outputs to the desired value during sleep mode. However, since the combinational circuit being analyzed is assumed to be part of a sequential network, the latches in the sequential network can be easily modified to force either a 0 or 1 during sleep mode. Fig. 2 shows standard static "jamb" latches [6] modified to force a value at the output during sleep mode. Fig. 3 shows modified dynamic C^2MOS flip-flops [7]. Clearly, other types of latches and flip-flops can also be modified in similar ways to force appropriate values during sleep mode.

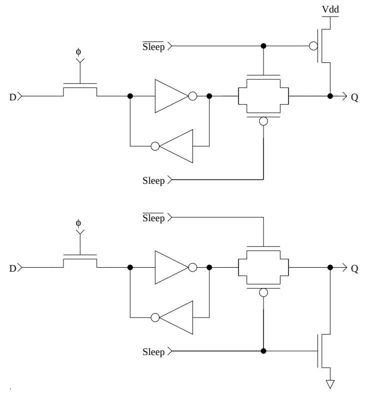

Figure 2. Modified "Jamb" Latches.

6. Experimental Results

We have tested this design methodology on the ISCAS-89 benchmark circuits. Table 2 shows the savings realized by this method using the probabilistic technique developed in Section 4 for error tolerances of 5% and 1% and also using a fixed, large sampling of 100,000 input vectors. These "savings" represent the percentage difference between the lowest leakage power obtained in each case and the largest leakage found for each circuit during the 100,000 vector run, relative to this largest leakage. Thus, one may think of these as being potential, or best case, savings. This method shows savings of up to 54% on circuits of various sizes.

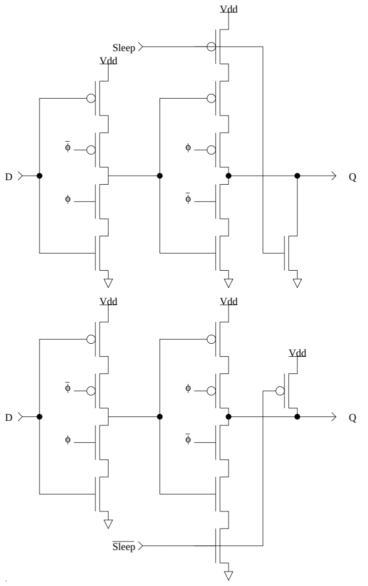

Figure 3. Modified C^{2}MOS Latches.

7. Conclusion

In this paper, we have proposed a novel design method that can be used during logic design to reduce the leakage power of CMOS circuits that use clock-gating to reduce the dynamic power dissipation. Using minimal additional circuitry, we modify the original logic design to force the combinational logic into a low-leakage state during an idle period. To find such a low-leakage state, we have developed an efficient algorithm that determines a good input vector using a sampling of random vectors. The size of this sampling is determined a priori using user-supplied quality measures. We have demonstrated this method on the ISCAS-89 benchmark circuits and shown leakage power reductions of up to 54%.

References

[1] A. Chandrakasan, et al., "Design considerations and tools for low-voltage digital system design," in *Proc. 33rd. Design Automation Conference*, pp. 113-118, 1996.

[2] G. E. Tellez, et al., "Activity-driven clock design for low power circuits,' in *Proc. IEEE Intl. Conf. Computer-Aided Design*, pp. 62-65, 1995.

[3] M. Horiguchi, et al., "Switched-source-impedance CMOS circuit for low standby subthreshold current giga-scale LSI's," *IEEE J. Solid-State Circuits*, vol. 28, pp. 1131-1135, Nov. 1993.

[4] S. Mutoh, et al., "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE J. Solid-State Circuits*, vol. 30, pp. 847-853, Aug. 1995.

[5] K. C. Kapur and L. R. Lamberson, *Reliability in Engineering Design*, New York, NY: Wiley, 1977.

[6] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, 2nd. Ed., Reading, Mass.: Addison-Wesley, 1992.

[7] J. M. Rabaey, *Digital Integrated Circuits: A Design Perspective*, Upper Saddle River, NJ: Prentice-Hall, 1996.

Table 2.
Power savings on ISCAS-89 benchmark circuits.

Benchmark Circuit	Gates	5% Savings	1% Savings	100k Savings
s1196	529	28.79%	28.57%	32.57%
s1238	508	30.93%	31.20%	34.47%
s13207	7951	7.80%	8.53%	9.80%
s1423	657	19.78%	20.39%	30.01%
s1488	653	22.53%	23.96%	25.81%
s1494	647	24.18%	24.68%	25.10%
s15850	9772	7.67%	8.36%	9.87%
s208	104	35.89%	40.85%	46.33%
s298	119	47.29%	52.76%	56.72%
s344	160	31.32%	37.58%	42.32%
s349	161	30.67%	32.72%	43.17%
s35932	16065	5.73%	6.21%	6.96%
s382	158	41.44%	40.50%	43.75%
s38417	22179	5.05%	5.77%	6.92%
s38584	19253	11.65%	12.37%	12.89%
s386	159	48.37%	50.91%	54.33%
s400	164	36.73%	38.74%	41.66%
s420	218	30.34%	35.46%	40.53%
s444	181	33.04%	33.29%	34.75%
s510	211	26.35%	28.10%	31.55%
s526	193	49.28%	51.44%	54.74%
s526n	194	44.48%	54.11%	55.25%
s5378	2779	5.93%	6.42%	6.96%
s641	379	34.77%	37.04%	39.56%
s713	393	32.00%	35.60%	38.26%
s820	289	38.02%	40.27%	42.75%
s832	287	37.86%	40.74%	43.12%
s838	446	26.83%	27.33%	32.61%
s9234	5597	8.36%	10.68%	10.96%
s953	395	28.92%	29.55%	36.98%